VŠB – Technická univerzita Ostrava
Fakulta strojní
Katedra automatizační techniky a řízení

Vizualizacní software pro monitorování a konfiguraci systému s PLC

Visualization Software for Monitoring and Configuration of system with PLC

Student: Bc. Pavel Šofer
Vedoucí diplomové práce: prof. Ing. Jiří Tůma, CSc.

Ostrava 2011

VŠB - Technická univerzita Ostrava
Fakulta strojní
Katedra automatizacní techniky a rízení

Zadání diplomové práce

Student:	Bc. Pavel Šofer
Studijní program:	N2301 Strojní inženýrství
Studijní obor:	3902T004 Automatické řízení a inženýrská informatika
Téma:	Vizualizační software pro monitorování a konfiguraci systémů s PLC Visualization Software for Monitoring and System Configuration with PLC

Zásady pro vypracování:

1. Seznamte se s prostředím pro programování PLC firmy ABB, včetně softwaru pro vizualizaci, a s možnostmi poskytování dat vyšší úrovni řízení.

2. Vytvořte návrh laboratorní úlohy realizující polohování vrtačky ve třech osách s ohledem na možnosti řídicího systému (PLC firmy ABB).

3. Realizujte části algoritmů umožňující polohování vrtačky s ohledem na zadaná data polohy např. formou tabulky.

4. Celou úlohu monitorujte ve zvoleném vizualizačním softwaru a vytvořte rozhraní pro konfiguraci reálné laboratorní úlohy.

5. Kriticky zhodnoťte dosažené výsledky a navrhněte další směr řešení.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. Ing. Jiří Tůma, CSc.**

Konzultant diplomové práce: Ing. Jaromír Škuta, Ph.D.

Datum zadání: 17.12.2010
Datum odevzdání: 23.05.2011

prof. Ing. Jiří Tůma, CSc.
vedoucí katedry

prof. Ing. Radim Farana, CSc.
děkan fakulty

Místopřísežné prohlášení studenta

Prohlašuji, že jsem celou diplomovou práci včetně příloh vypracoval samostatně pod vedením vedoucího diplomové práce a uvedl jsem všechny použité podklady a literaturu.

V Ostravě : 23. 5. 2011

.....
Bc. Pavel Šofer

Prohlašuji, že

- jsem byl seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, zejména §35 – Užití díla v rámci občanských či náboženských obřadů nebo v rámci úředních akcí pořádaných orgány veřejné správy, v rámci školních představení a užití díla školního a §60 – Školní dílo.
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen „VŠB-TUO“) má právo nevýdělečně ke své vnitřní potřebě diplomovou práci užít (§35 odst. 3 autorského zákona).
- souhlasím s tím, že jeden výtisk diplomové práce bude uložen v Ústřední knihovně VŠB-TUO k prezenčnímu nahlédnutí a jeden výtisk bude uložen u vedoucího diplomové práce. Souhlasím s tím, že údaje o diplomové práci, obsažené v Záznamu o závěrečné práci, umístěném v příloze mé diplomové práce, budou zveřejněny v informačním systému VŠB-TUO.
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu §12 odst. 4 autorského zákona.
- bylo sjednáno, že užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

V Ostravě: 23.5.2011

.....

Bc. Pavel ŠOFER
Svornosti 1656/17
748 01 Hlučín

ANOTACE DIPLOMOVÉ PRÁCE

ŠOFER, P. *Vizualizační software pro monitorování a konfiguraci systémů s PLC.* Ostrava: katedra ATŘ – 352 VŠB-TUO, 2011, 66s. Diplomová práce, vedoucí: Tůma, J., konsultant: Škuta, J.

Tato diplomová práce se zabývá návrhem a realizací laboratorní úlohy, která umožňuje polohování stolního modelu CNC stroje ve třech osách. Stolní model CNC stroje zde slouží jako nástroj k odvrtávání pájecích děr na deskách plošných spojů, navržených v programu EAGLE.

Prostředníka mezi programem EAGLE a CNC strojem zde představuje monitorovací a vizualizační aplikace, vytvořená v programu Control Web 6, která s využitím DDE komunikačního protokolu přijímá vyexportovaná data z programu EAGLE, vhodnými algoritmy je převádí na potřebné souřadnice pájecích otvorů a následně odesílá tato data prostřednictvím OPC protokolu PLC automatu, který ovládá model CNC stroje.

Součástí této diplomové práce je podrobný popis vytvořených aplikací, algoritmů a hardwarových unifikačních obvodů.

ANOTION OF THE DIPLOMA THESIS

ŠOFER, P. *Visualization Software for Monitoring and Configuration of system with PLC*. Ostrava: Department of Control Systems and Instrumentation, VŠB-Technical University of Ostrava, 2011, 66 p. Thesis supervisor: Tůma, J., Thesis consultant: Škuta, J.

This diploma thesis deals with design and realisation of a laboratory task which is focused on 3-axis positioning of a CNC machine model. This desktop CNC machine is used here as a tool for solder holes drilling of a PCB, which was designed using the EAGLE software.

Connection between the EAGLE software and the CNC machine model provides an monitoring and visualisation application, created in the Control Web 6 system. This application receives exported data from the EAGLE software via a DDE communication protocol and using a suitable algorithms transfers these data to a desirable solder holes coordinates. These coordinates are then sent to a PLC controller which controls the CNC machine via the OPC communication protocol.

Detailed description of created algorithms, application and a hardware unification circuits is also a part of this thesis.

Obsah

Seznam použitých zkratek

3DS	formát animačního souboru
AND	operátor logického součinu
CW5	systém Control Web 5
CW6	systém Control Web 6
CNC	počítačové číslicové řízení (Computer Numerical Control)
DDE	komunikační protokol (Dynamic Data Exchange)
DMF	mapovací soubor systému Control Web
FBD	programovací jazyk pro PLC - funkční diagram
IL	programovací jazyk pro PLC - instrukční list
LD	programovací jazyk pro PLC - funkční diagram
OLE	propojování a vkládání objektů (Object Linking and Embeddind)
OPC	standardní komunikační rozhraní prvků průmyslové automatizace (OLE for Process Control)
OR	operátor logického součtu
PAR	parametrický soubor systému Control Web
PLC	programovatelný logický automat (Programmable Logic Controller)
RTU	vzdálená terminálová jednotka
SCADA	supervizní řízení a sběr dat (Supervisory Control And Data Acquisition)
SFC	programovací jazyk pro PLC - sekvenční funkční diagram
ST	programovací jazyk pro PLC - strukturovaný text
UNICODE	mezinárodní kódovací standard
XLS	přípona souborů aplikace Microsoft Excel
XOR	operátor exkluzívního logického součtu

1 Úvod

V dnešní době je řízení neodmyslitelnou součástí našeho denního života. Většina námi denně používaných věcí, které považujeme za naprostou samozřejmost, prošla technologickými procesy výroby, které mohly být velmi jednoduché nebo naopak vysoce sofistikované a složité. Vysoká složitost těchto procesů s sebou obvykle nese i požadavek na co nejkvalitnější řízení takovéhoto procesu, aby byla zaručena požadované jakost a samozřejmě z ekonomického hlediska i co nejnižší zmetkovitost. Nelze zanedbat fakt, že se složitostí výroby daného produktu, rostou náklady na jeho výrobu, což se v konečném důsledku projeví v jeho ceně.

Pokud bychom se u výrobních procesů soustředili pouze na proces řízení, mohli bychom slepě důvěřovat, že systém je navržen a pracuje správně. Mohli bychom například data ze senzorů řízeného procesu shromažďovat v tabulkových formách, avšak u rozsáhlejších procesů by se tento způsob brzy stal nepřehledným a téměř zbytečným. Mnoho vědeckých studií, zaměřených na způsoby vstřebávání informací u člověka prokázalo, že lidé jsou schopni nejefektivněji přijímat informace v názorné obrazové formě. Proto je důležitou součástí většiny technologických procesů i vizualizace. Vizualizace je účinný nástroj, jak co nejefektivněji sdělovat důležité informace v obrazové formě o řízeném procesu tak, aby byla zajištěna co nejlepší zpětná vazba mezi procesem a uživatelem, potažmo operátorem. Dnešní vizualizační systémy neumožňují procesy pouze monitorovat, ale lze prostřednictvím nich do procesů přímo zasahovat. Mezi rozsáhlou skupinu vizualizačních systémů patří i program Control Web 6. V této diplomové práci využívám tento vizualizační software pro monitorování a vizualizaci laboratorní úlohy, která umožňuje polohování stolního modelu CNC stroje. Aplikace vytvořená v systému Control Web 6 obsahuje algoritmy pro zpracovávání dat a zároveň prostřednictvím DDE standardu komunikuje s programem EAGLE a odesílá data do PLC, využitím OPC protokolu.

2 Vizualizační software Control Web 6

V oblasti vizualizačních softwarů figuruje celá řada produktů, které se liší svým zaměřením, vlastnostmi, schopnostmi, ale také zejména cenou. Mezi tyto produkty patří i software Control Web 6, vytvořený zlínskou firmou Moravské přístroje a.s. Systém Control Web 6 patří do skupiny tzv. SCADA systémů (Supervisory Control And Data Acquisition). Hlavní funkcí každého SCADA systému je sběr dat a informací a zároveň poskytnutí rozhraní pro řízení určitého zařízení. Tímto zařízením může být například PLC, RTU atp. SCADA systémy umožňují neustálý dohled a kontrolu nad řízenými procesy v nejrůznějších průmyslových odvětvích, jako například elektrárenství, potravinářství, petrochemii.

Systém Control Web 6 vychází z osvědčené architektury svého předchůdce - Control Web 5. Control Web 6 lze rovněž definovat jako:

- Programový systém rychlého vývoje aplikací pro průmysl, laboratoře, školy, atp.
- Vizualizace a řízení technologických procesů v reálném čase.
- Most mezi technologií a informačním systémem podniku.
- Rozhraní člověk-stroj.
- Přímé řízení strojů a technologií.
- Simulace, výzkum, vývoj a výuka.

Jednou z mnoha předností systému je jeho schopnost provozovat modulární, distribuované a zálohované aplikace:

- *Modulární* je aplikace, která sestává z více modulů. Control Web považuje za modul každý jeden aplikační soubor, je tedy možné říci, že nejjednodušší aplikace je jedno-modulární, taková aplikace se celá nachází v jediném zdrojovém souboru.
- *Distribuovaná* je aplikace, ve které jednotlivé její sdílené součásti běží na různých počítačích. Distribuovaná aplikace v systému Control Web vždy běží v módu klient-server.
- *Zálohovaná* je v podstatě distribuovaná aplikace, ve které jsou některé její sdílené součásti udržovány v synchronním stavu (obsahují v jednom okamžiku stejná data). Zatímco nezálohovaná distribuovaná aplikace má přesně stanoveno, která součást musí být server a která klient, zálohovaná aplikace toto stanoveno nemá. V každém okamžiku je tedy možné říci, že nyní je tento modul server a tento modul klient nebo naopak [Moravské přístroje 2006].

2.1 Vývojové prostředí systému Control Web 6

Vývoji vizualizační aplikace vždy předchází přesné určení požadavků na funkci a schopnosti dané aplikace. I když spousta vizualizačních softwarů obsahuje rozsáhlé grafické knihovny pro usnadnění tvorby aplikace, ne vždy si vývojář vystačí pouze s nimi a tvorba takové aplikace pak bývá časově velmi náročná a v praxi tudíž velmi nákladná. Je proto potřeba předem znát a jasně definovat strukturu a zaměření této aplikace.

Po stanovení těchto požadavků, nejčastěji ze strany zákazníka, se však často vyskytne otázka, jakým způsobem tuto aplikaci vytvořit. Některým z nás stačí k jasnému pochopení určitého problému pouhý popis všech vlastností, avšak jiným je zapotřebí daný problém graficky znázornit. Stejně jako u svého předchůdce, systém CW6 rovněž využívá při vývoji aplikací tzv. dvojcestné programování. Jedná se o dvě varianty vývojového prostředí, grafický editor a textový editor, které jsou navzájem propojeny a každá změna při vývoji aplikace v jednom z nich se automaticky projeví do druhého. Tento proces se nazývá překlápění. Z uvedeného popisu vyplývá, že obě formy zápisu aplikace (grafická i textová) jsou navzájem zástupné. Vzhledem k tomu, že CW6 je systém hierarchický, bylo nutné jednu formu zápisu aplikace upřednostnit před druhou a to formu textovou. Toto rozhodnutí ze strany vývojářů programu je pochopitelné. Grafická forma uchovávání dat obecně bývá často velmi složitá a paměťově náročná. Naopak textový zápis byl historicky první způsob uchovávání dat a i v systému CW6 je velmi snadné aplikaci, převedenou do textové formy, uchovávat a jakkoliv s ní manipulovat. Zdrojový tvar jakékoliv aplikace vytvořené v systému CW6 je proto textový.

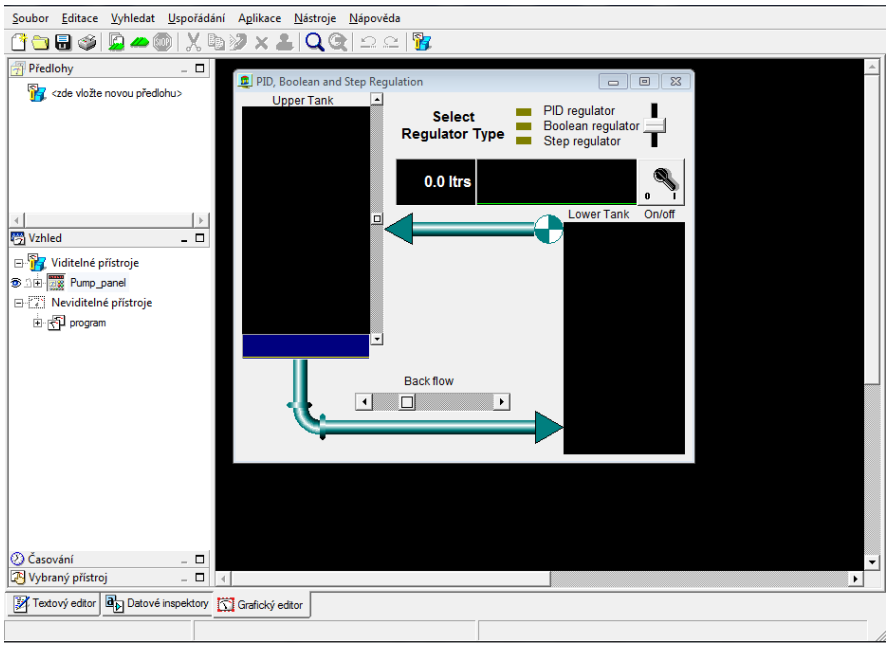
Obr. 1 Překlápění aplikace

Jak znázorňuje Obr. **1**, při převodech mezi textovými a grafickými režimy aplikace dochází k tzv. překlápění aplikace, což se může zdát jako jedna činnost, nehledě na to z/do kterého režimu aplikace se přepínáme. Ve skutečnosti však dochází ke dvěma různým operacím, kdy při převodu aplikace z textového na grafický režim dojde k tzv. překladu a při převodu aplikace z grafického na textový režim dochází k tzv. generování. Při tvorbě aplikace v textovém editoru je možné zapsat prakticky jakýkoliv text. Na problém však narazíme, kdybychom takovou aplikaci chtěli přeložit a pracovat s ní v grafickém editoru. CW6 totiž striktně vyžaduje přesný tvar zápisu aplikace, v opačném případě vyvolá při překladu chybu.

2.1.1 Grafický editor

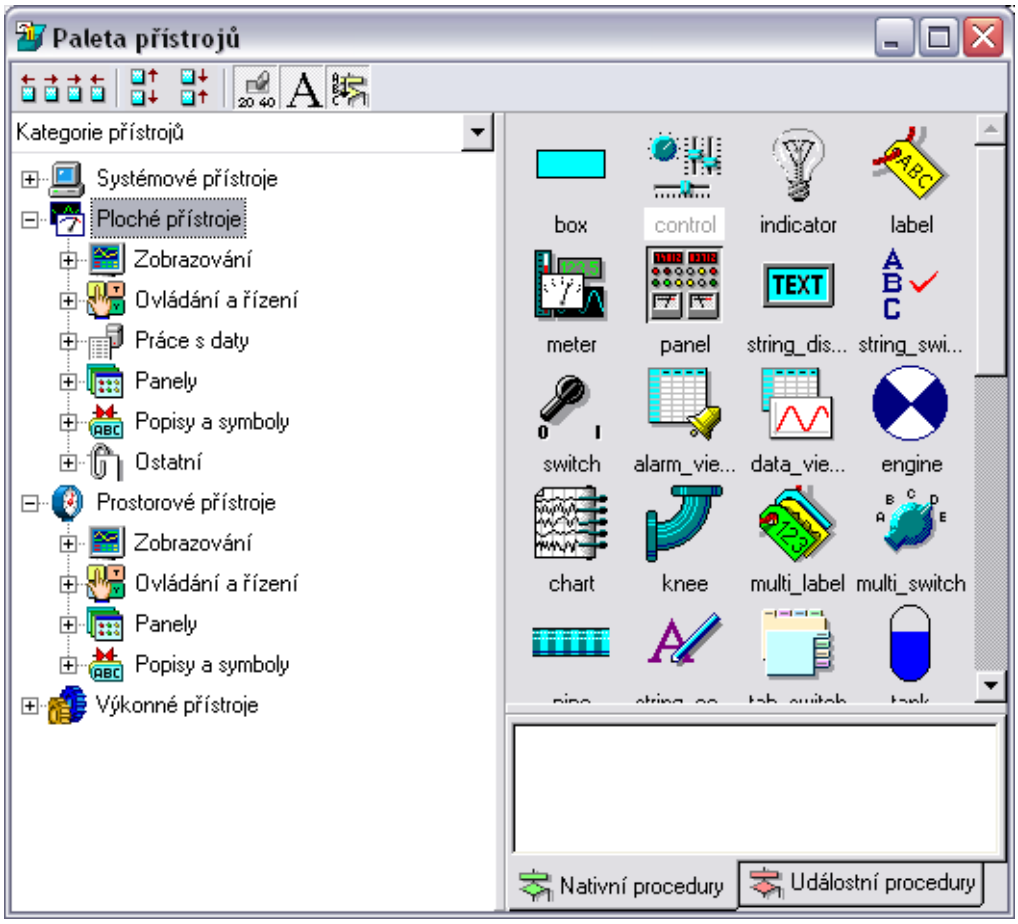
Velmi důležitou součástí vývoje každé aplikace je bezpochyby její vzhled. V praxi se často stává, že vizualizované technologické procesy musí být neustále pod dohledem člověka na operátorském pracovišti. Takovéto vizualizační aplikace pak musí být navrženy s určitou dávkou ergonomie a s ohledem na lidské zdraví. Volba barev, rozmístění prvků na obrazovce a další důležité faktory vzhledu je třeba zohlednit tak, aby konečná podoba aplikace měla co nejlepší dopad na lidské zdraví, ale také na efektivitu zobrazovaných informací.

Grafický editor se skládá z několika logicky rozdělených částí. Jedna bez druhé by byla nesmyslná a nefunkční, ale pro účel popisu je moudré celý grafický editor rozdělit a popsat jednotlivé části postupně. Komunikaci s reálným světem, použitá data, vnitřní údaje a nastavení parametrů aplikace shrnují Datové inspektory. Nabídku přístrojů, spolu s jejich základním popisem (například procedur) obsahuje Paleta přístrojů. Prostor pro vizuální návrh a řízení běhu aplikace pokrývá hlavní část vývojového prostředí, vlastní Grafický editor. Každý objekt lze rychle změnit a upravit v Inspektoru přístroje [Moravské přístroje 2006].



Obr. 2 Grafický editor

Grafický editor si lze rovněž představit jako jakýsi kontejner, do nějž vkládáme dle potřeby jednotlivé přístroje. Takto vložené přístroje lze jednoduše upravovat, měnit jejich velikost, typ použití, vázat na ně proměnné a mnoho jiných dalších operací. Stejně jako potřebujeme místo kam zvolené přístroje vkládat, potřebujeme i místo, kde tyto přístroje organizovaně shromažďovat. Takovým místem je v systému CW6 Paleta přístrojů.



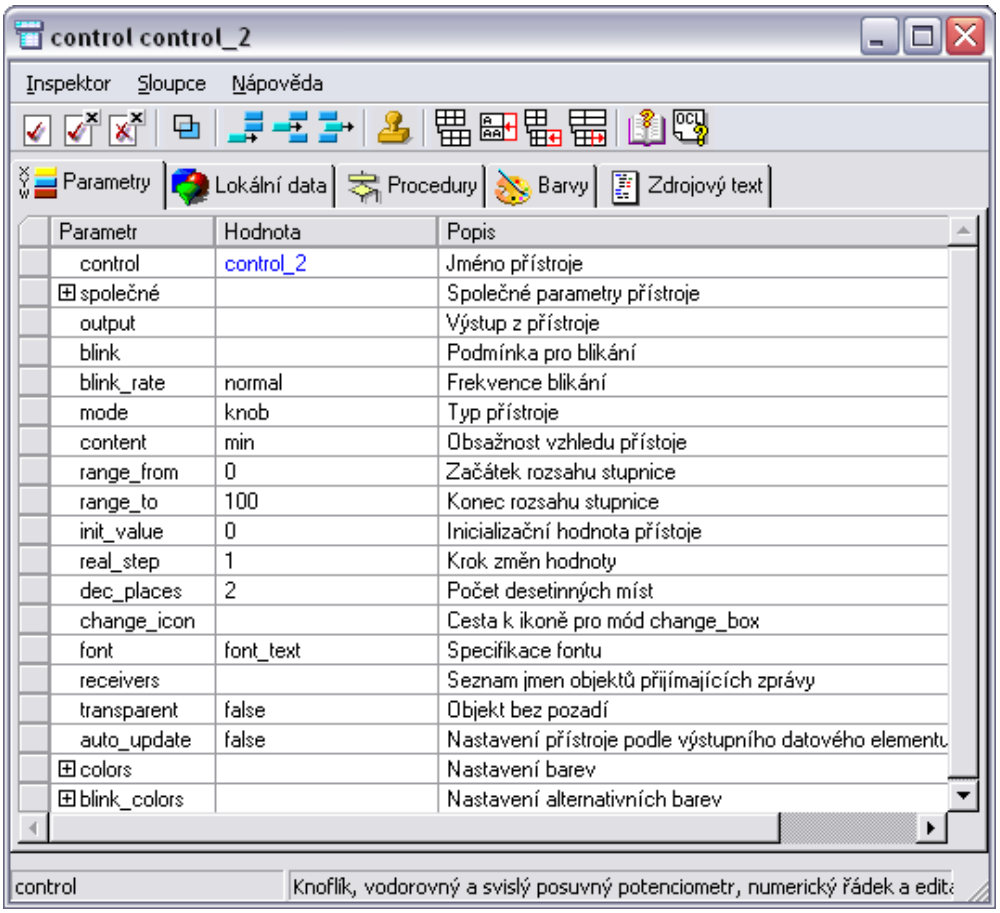
Obr. 3 Paleta přístrojů

Strom s kategoriemi v pravé části obsahuje všechny dostupné přístroje, které je do grafického editoru možné přímo přetahovat myší, rozčleněné podle své funkce a použití. Pomocí výběrového seznamu nad stromem lze rovněž strom přepnout pro zobrazení knihoven přístrojů neboli přístrojů rozčleněných podle jejich příslušnosti ke knihovnám. Každý řádek stromu má svou funkci a omezuje, je-li vybrán, množství přístrojů, které jsou nabízeny v pravé části palety, v ploše s ikonami. Kategorie a podkategorie zapsané ve stromu jsou členěny stále přesněji až na konečnou úroveň jednotlivých přístrojů v kategorii. Vyberete-li ve stromu takový řádek s přístrojem, naplní se plocha s ikonami buď jedinou ikonou přístroje (což platí třeba pro přístroj program) nebo vlastní paletou přístroje, kde přístroj nabízí svá předpřipravená nastavení (opět pro snadné a rychlé použití). Vlastní paletu mají především přístroje, které slouží k vizualizaci nebo přístroje, které slouží v aplikaci pro vytváření vzhledu a grafiky [Moravské přístroje 2006].

2.1.2 Inspektor přístroje

Inspektor přístroje je jeden ze základních nástrojů grafického editoru, pomocí něhož dokážeme editovat jakýkoliv vložený přístroj. Můžeme upravit jeho parametry, přidávat procedury, navazovat proměnné a jiné důležité akce. Většina vložených přístrojů se však liší svým typem - zobrazovací, řídicí, apod. a proto je nutné, aby pro každý takový typ přístroje byl vlastní inspektor. Tyto se však liší pouze několika parametry v závislosti na typu přístroje.

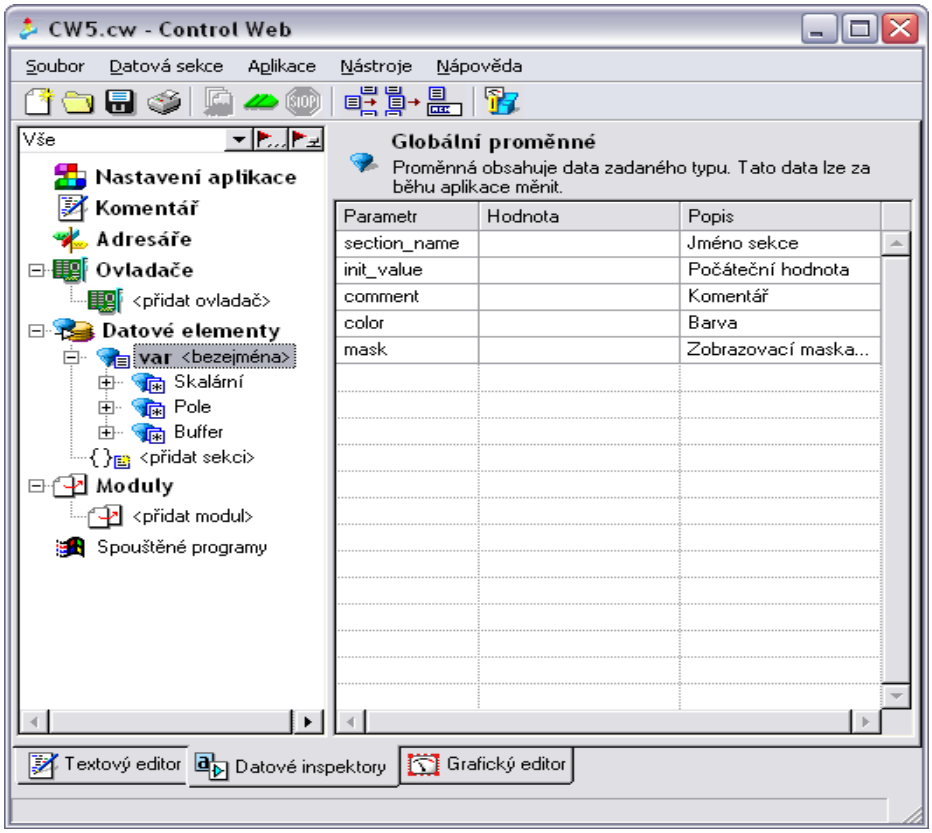
Nově otevřený inspektor vždy obsahuje právě aktuální data přístroje. Data lze dle potřeby upravovat, přičemž veškeré úpravy probíhají pouze uvnitř inspektoru a neovlivňují vlastní editovaný přístroj. Editaci přístroje v inspektoru lze ukončit dvěma způsoby — buď použitím úprav nebo zrušením úprav. Použití zapíše údaje upravené v inspektoru do editovaného přístroje, zrušení jen zavře okno inspektoru a editovaný přístroj nijak neovlivní. Jakoukoli změnu dat přístroje je tedy v inspektoru nutné potvrdit (použitím), jinak se v přístroji žádná změna neprojeví [Moravské přístroje 2010].



Obr. 4 Inspektor přístroje

2.1.3 Datový inspektor

V předchozí kapitole byl popsán grafický editor, jehož produktem jsou přístroje. Aby však tyto přístroje byly využity, je třeba jim poskytnou datové elementy, se kterými budou pracovat. Účinným nástrojem, jak tyto elementy vytvářet a spravovat je datový inspektor.



Obr. 5 Datový inspektor

Datové proměnné, vytvořené v datovém inspektoru se ukládají vždy jako globální. Výhodou tohoto je, že jsou dostupné všem přístrojům v aplikaci a jejich procedurám.

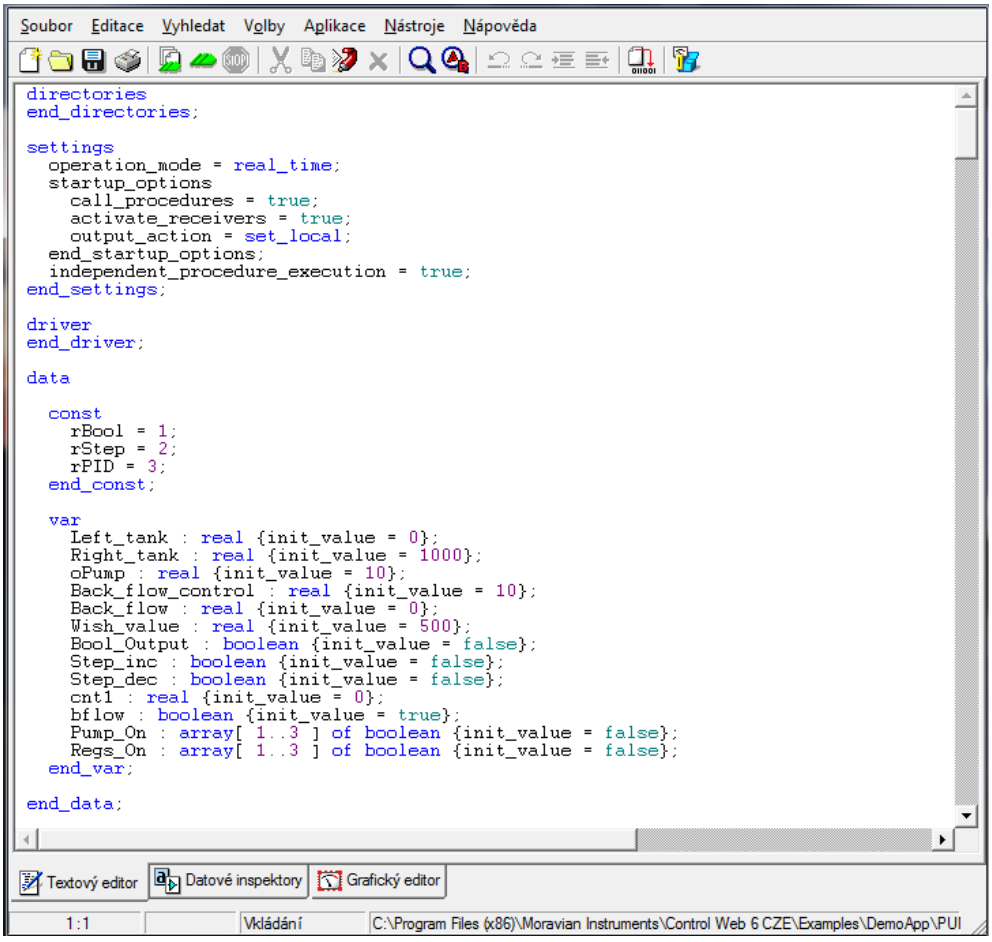
Datové elementy slouží k uchovávání dat, kterých Control Web rozeznává čtyři hlavní skupiny [Bílý a Cagaš 1999]:

- Logické hodnoty obsahují binární hodnotu vyjadřující stav zapnuto/vypnuto (true/false).
- Číselné hodnoty, jsou desetinné (real, shortreal) nebo celé čísla (shortint, shortcard, ineger, cardinal, longint, longcard), a to buď se znaménkem (real, shortreal, shortint, ineger, longint) nebo bez znaménka (shortcard, cardinal, longcard).
- Řetězce znaků (string), které obsahují skupiny písmen (libovolné kromě znaku s hodnotou 0) bez omezení délky, které se zapisují do apostrofů (").
- Datový typ buffer, obsahující blok paměti.

2.1.4 Textový editor

Jak již bylo uvedeno dříve, celou aplikaci lze navrhnout i v textovém editoru. Textový editor systému Control Web pracuje s libovolnými textovými soubory. Kromě toho dokáže pracovat i s textovými soubory se znaky ve formátu UNICODE, ale tato schopnost je omezena pouze na systémy, které UNICODE podporují (v současné době Windows NT, Windows 2000 a Windows CE). Textový editor má rozšíření, která jsou šita na míru systému Control Web, nicméně jeho základní funkčnost je zcela obecná a umožňuje úpravy jakéhokoli textu. Základní principy ovládání dodržují konvence editorů grafického uživatelského rozhraní operačních systémů Windows [Moravské přístroje 2010].

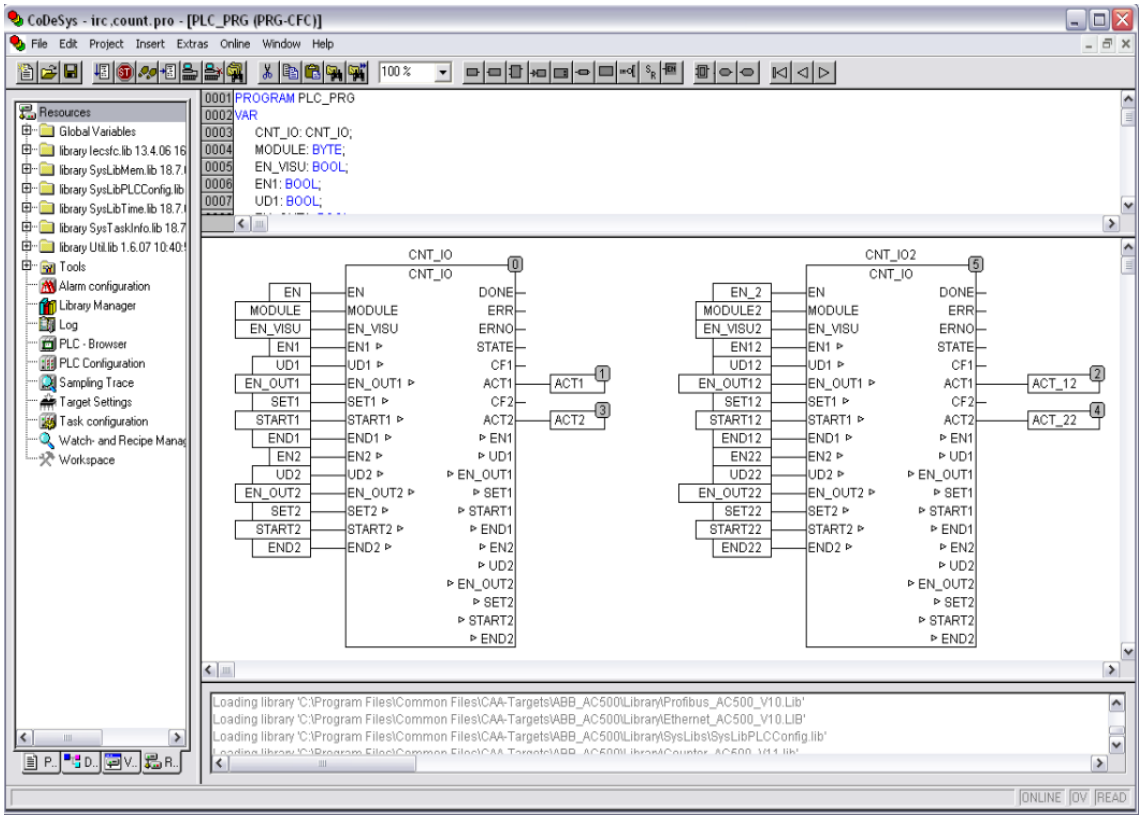
Aplikace vytvořená v textovém editoru je sice velmi snadno přenositelná a upravitelná, naopak její vytvoření vyžaduje velmi dobrou znalost tvorby aplikací v systému CW6 neboť i ta nejbanálnější chyba neumožní spuštění aplikace. Následující obrázek znázorňuje část aplikace převedené z grafického do textového režimu. V úvodu kódu je zobrazeno startovací nastavení celé aplikace a obsažené datové elementy. Z obrázku je patrné, že textový režim aplikace je velmi přehledný a struktura řazení i syntaxe jednotlivých částí se velmi podobá jazyku C++.



Obr. 6 Textový editor

2.2 Vývojové prostředí pro PLC

Při automatizaci technologických procesů je nutné zajistit jejich spolehlivou funkčnost a pokud možno co nejnižší poruchovost. Jednou z možností jak tohoto docílit je použití PLC, programovatelných logických automatů. Jsou konstruovány tak, aby zajistily spolehlivý provoz ve velmi těžkých podmínkách. Softwarů, které umožňují programovat PLC existuje celá řada. Mnoho výrobců PLC k nim dodává i vlastní programovací software, který je navržen na konkrétní modelovou řadu k dosažení maximální efektivity při tvorbě a nahrávání programů do PLC. V této diplomové práci jsem využíval program CoDeSys, produkt německé firmy 3S - Smart Software Solutions, založené v roce 1994. Jedná se o komplexní nástroj pro programování a vizualizaci procesů, řízených pomocí PLC. Při tvorbě aplikací v tomto softwaru je vývojáři k dispozici několik programovacích jazyků, které je možné mezi sebou kombinovat a vytvářet tak uživatelem definované funkční bloky a makra, které podstatně usnadňují další práci.



Obr. 7 Prostředí programu CoDeSys

Programovací jazyky pro PLC

Při tvorbě aplikace pro PLC v programu CoDeSys je na výběr z těchto programovacích jazyků:

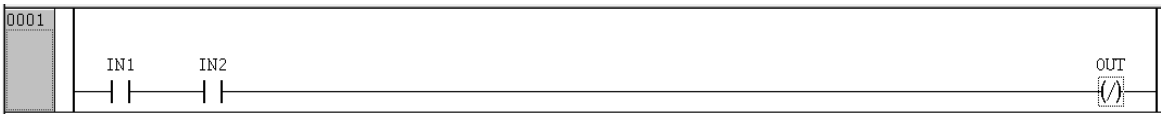
Instrukční list – IL (Instruction list)

Programovou organizační jednotku tvoří sekvence instrukcí, z nichž každá začíná na novém řádku a může obsahovat návěští, ukončené dvojtečkou, operátor, který může být případně doplněn tzv. modifikátorem, operand a někdy také komentář. Pomocí modifikátorů se vyjadřují negace, podmíněnost a nepodmíněnost instrukce skoků, volání a návratů a priorita. Příklad části programu napsaného v jazyce IL [Martinásková 2004]:

```
LD 17
ST lint (* Kommentar *)
GE 5
JMPC next
LD idword
EQ istruct.sdword
STN test
next:
```

Příčkový diagram – LD (Ladder diagram)

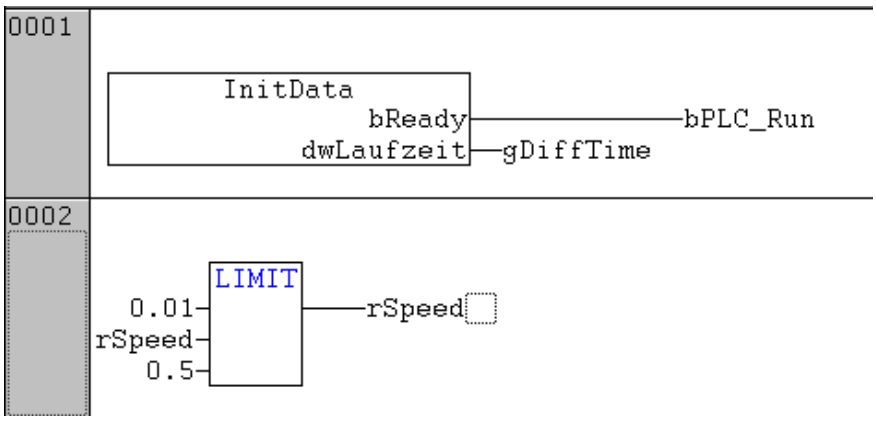
Grafický jazyk LD je někdy také nazýván jazykem kontaktních schémat a je založen na grafické reprezentaci reléové logiky. Organizační jednotka programu je vyjádřena sítí propojených grafických prvků. Síť v jazyku LD je zleva i zprava ohraničena svislými čarami, které se nazývají levá a pravá napájecí sběrnice. Mezi nimi je tzv. příčka, která může být rozvětvena. Každý úsek příčky, vodorovný nebo svislý, může být ve stavu on nebo off. Do příček mohou být včleněny kontakty (spínací, rozpínací apod.), cívky (cívka, negovaná cívka apod.) a dále funkce a funkční bloky [Martinásková 2004].



Obr. 8 Příčkový diagram

Funkční diagram – FBD (Function Block Diagram)

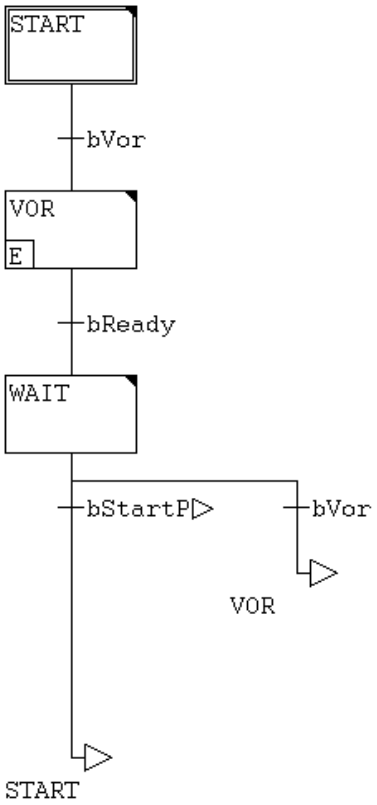
Jazyk FDB vyjadřuje chování funkcí, funkčních bloků a programů jako soubor vzájemně provázaných grafických bloků podobně jako v elektronických obvodových diagramech. Jde o systém prvků, které zpracovávají signály. Často se zde používají standardní funkční bloky, jako jsou např. bistabilní prvky, prvky pro detekci náběžné a sestupné hrany, čítače apod [Martinásková 2004].



Obr. 9 Funkční diagram

Sekvenční funkční diagram – SFC (Sequential Function Chart)

SFC popisuje sekvenční chování řídicího programu. Je odvozen ze symboliky Petriho sítí, ale liší se od nich tím, že grafická reprezentace se zde převádí přímo do souboru výkonných řídicích prvků. SFC strukturalizuje vnitřní organizaci programu a umožňuje rozložit úlohu řízení na zvládnutelné části a zachovat přitom přehled o chování celku [Martinásková 2004].

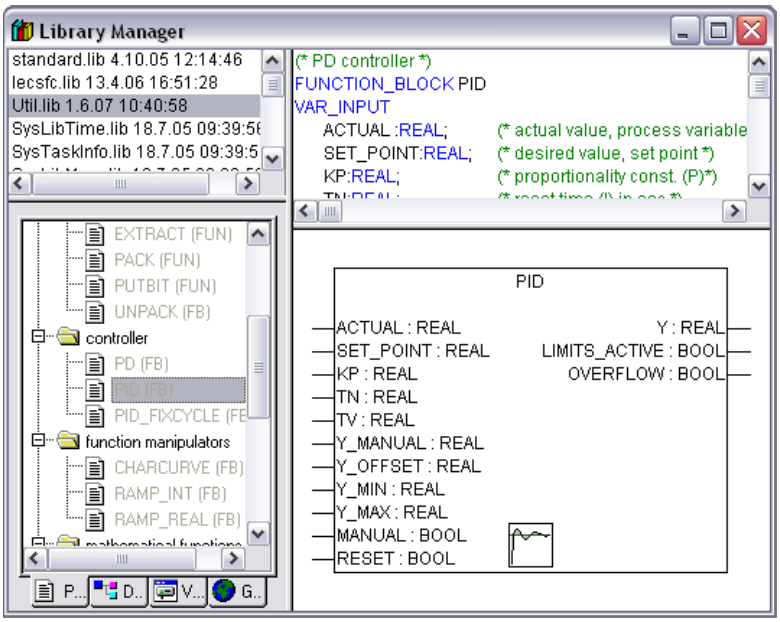


Obr. 10 Sekvenční funkční diagram

Strukturovaný text – ST (Structured Text)

ST je výkonný vyšší programovací jazyk, který má kořeny v jazycích Pascal a C. Syntaxe jazyka je dána povolenými výrazy a příkazy. Vyhodnocením výrazu vyjde hodnota v některém z definovaných datových typů. Výraz se skládá z operátorů a operandů. Operandem může být konstanta, proměnná, funkce nebo jiný výraz. Operátory pro jazyk ST jsou definovány pro sedmnáct typů operací (vyhodnocení funkce, negace, násobení, booleovské funkce AND, XOR a OR apod.). Je definováno deset typů příkazů (přiřazení, vyvolání funkce, návrat, výběr apod.). Příkazy jsou odděleny středníkem a může jich být více na jednom řádku. Jazyk ST je vhodným nástrojem pro definování komplexních funkčních bloků, které pak mohou být použity v libovolném programovacím jazyku [Martinásková 2004].

Velkou oporu a ulehčení práce také představuje tzv. knihovna bloků. Jedná se o rozsáhlou knihovnu předdefinovaných funkčních bloků, která už v základní verzi čítá velké množství prvků, počínaje nejjednoduššími funkcemi AND, až po složité funkční bloky rychlých čítačů, PID regulátorů a podobně.



Obr. 11 Knihovna funkčních bloků

Při práci s funkčními bloky je velkou výhodou jednoduchost, s jakou s nimi lze manipulovat. Mohou se však vyskytnout situace, kdy by se tato jednoduchost mohla jevit naopak jako nevýhoda neboť každý funkční blok má pevně danou strukturu a vlastnosti, které nelze měnit. Jakýkoliv požadavek na úpravu chování daného bloku by znamenalo vytvoření podprogramu, zajišťujícího požadované vlastnosti nebo hledáním jiného řešení.

3 Návrh laboratorní úlohy s CNC modelem

Navrženou laboratorní úlohu s využitím CNC modelu názorně popisuje následující schéma. Základní myšlenkou je polohování stolního modelu CNC stroje podle dat, získaných z programu EAGLE a převedených do souřadnic X, Y.

Obr. 12 Schéma laboratorní úlohy s CNC modelem

Funkci laboratorní úlohy lze rozdělit do několika fází:

a) **Návrh desky plošných spojů**

V první fázi je nutné navrhnout desku plošných spojů ve vhodném programu. Programů pro návrh desek plošných spojů existuje celá řada. Pro tuto laboratorní úlohu byl vybrán program EAGLE, který je vhodný pro svou komplexní a přehlednou strukturu a možnosti exportování vytvořených schémat desek plošných spojů do různých formátů výstupního souboru.

b) **Vizualizační a výpočtová aplikace v CW6 pro převod a zpracování dat**

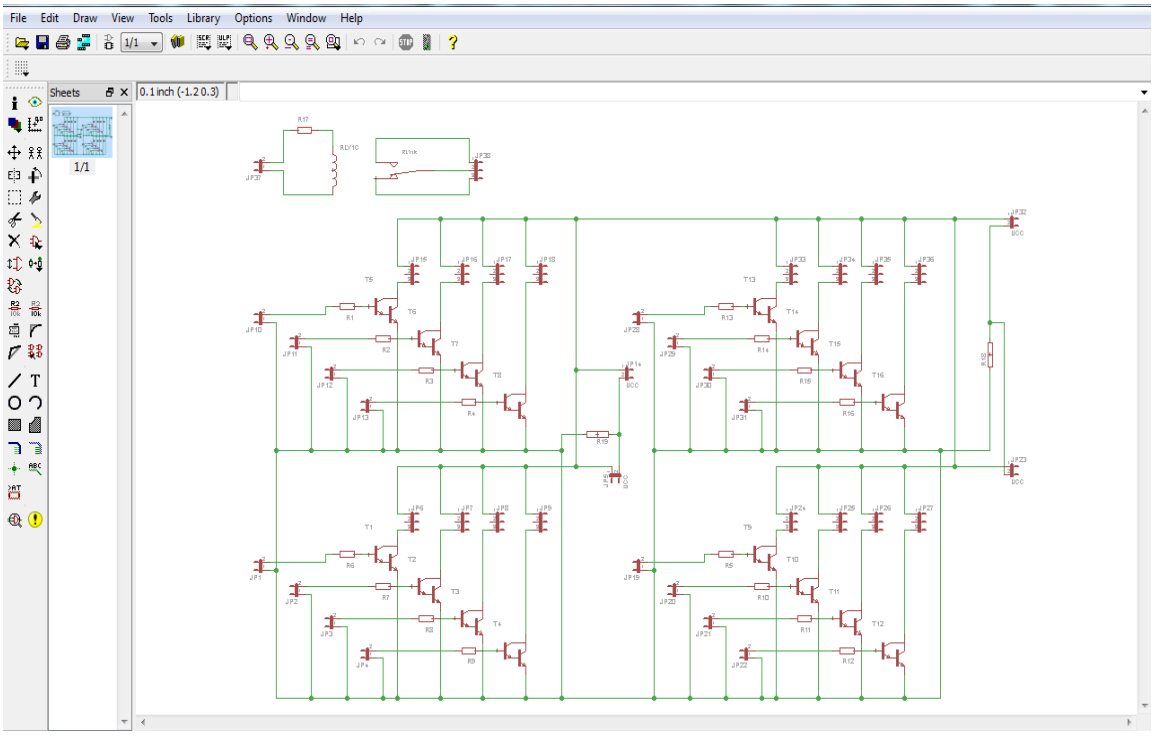
Jak bylo uvedeno výše, program EAGLE umožňuje různé druhy exportu výstupních dat. Jednou z těchto možností je export ve formě tzv. Partlistu, který je optimální pro tuto úlohu. Tento typ exportu ukládá do výstupního souboru, ve zvoleném formátu, informace o každé součástce, nacházející se na navržené desce. Každá součástka je popsána svojí polohou [X,Y] vzhledem k referenčnímu bodu na desce, dále pak případnou rotací, názvem pouzdra a knihovny, ve které je součástka obsažena. Tyto informace, vždy o každé součástce zvlášť, se zapisují do jednoho textového řetězce, se kterým se bude dále pracovat.

c) **Ovládací algoritmy v PLC**

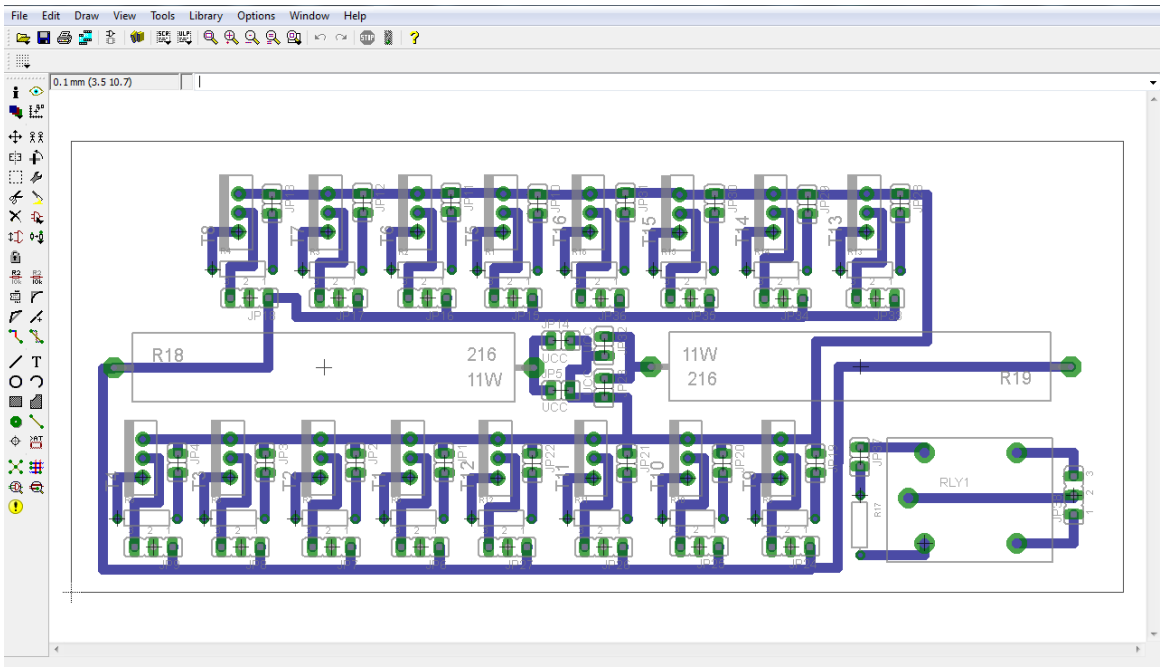
Zpracovaná data, která již obsahují jednotlivé souřadnice [X,Y] vrtacích otvorů, jsou odesílána ve formě polí proměnných do PLC. Komunikaci mezi systémem CW6 a PLC zajišťuje OPC protokol. Pro tuto úlohu jsem použil PLC firmy ABB, modelové řady 500.

3.1 Využití programu EAGLE

Program EAGLE (Easily Applicable Graphical Layout Editor) slouží převážně k návrhu elektrických zapojení a desek plošných spojů. Výhodou tohoto programu je, že v jeho základní verzi je program šířen jako freeware. Tato freeware verze je sice omezena velikostí desky a počtem vrstev, ne však počtem vložených součástek.

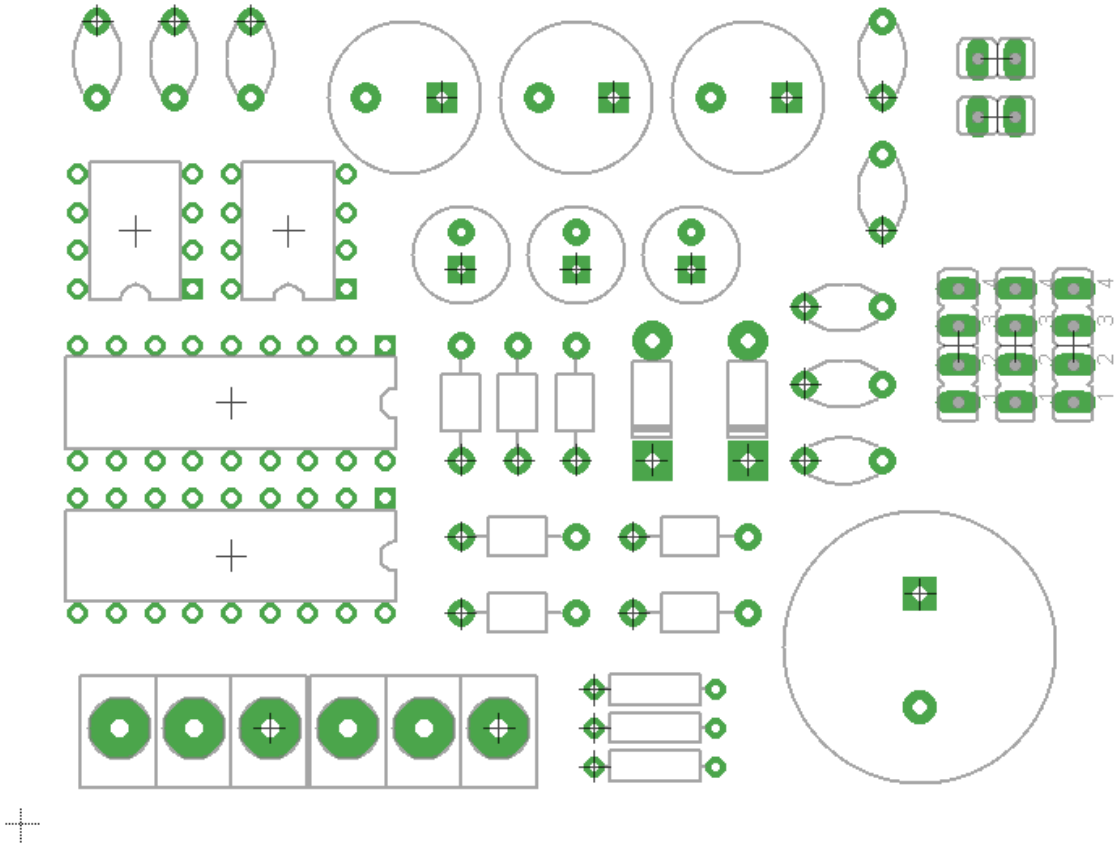


Obr. 13 Prostředí pro návrh a úpravu schémat zapojení programu EAGLE



Obr. 14 Prostředí pro návrh a úpravu desek plošných spojů programu EAGLE

Při návrhu desky plošných spojů v programu EAGLE se jednotlivé elementy (díry, signálové trasy, součástky atp.) vykreslují odlišnými barvami a překreslují v odpovídajících vrstvách. Každá vrstva má svůj význam, název a barvu, kterou vykresluje jednotlivé objekty. Tyto vrstvy je možno v programu EAGLE přehledně editovat a volit, které vrstvy budou na obrazovce zobrazeny a které nikoliv, viz. Obr. **15**.



Obr. 15 Příklad vykreslení zvolených vrstev

Na výše uvedeném obrázku je deska plošných spojů, kde jsou vykresleny jen zvolené grafické vrstvy. Jsou zde patrné tvary použitých součástek, jejich pájecí otvory, a také v levém dolním rohu velmi důležitý referenční bod desky [0,0].

Důležitou funkcí programu EAGLE je export informací o navržené desce do výstupního souboru. Program EAGLE nabízí několik variant exportu, z níž pro tuto úlohu je nejdůležitější export ve formě tzv. Partlistu. Tento typ výstupu udává informace o přesné poloze a orientaci součástek, umístěných na navržené desce plošných spojů. Jsou zde však uvedeny pouze ty součástky, které obsahují pájecí body.

Příklad zápisu do výstupního souboru při exportu ve formě Partlistu:

Partlist

Exported from KrokMot.brd at 3.1.2011 11:27:03

EAGLE Version 5.6.0 Copyright (c) 1988-2009 CadSoft

Part	Value	Package	Library	Position (mm)	Orientation
R1		RR	GM99	(54.03 42.8)	R0
R2		RR	GM99	(42.58 42.83)	R90
R3		RR	GM99	(30.76 42.8)	R270
R4		RR	GM99	(18.81 42.9)	R0
R5		RR	GM99	(90.96 9.8)	R90

Výstupní soubor obsahuje informace o názvu součástky, příslušném pouzdře, knihovně součástek programu EAGLE, ve které je daná součástka obsažena, přesnou pozici [X,Y] součástky vzhledem k referenčnímu bodu [0,0] a natočení součástky o daný úhel. I kdyby se mohlo zdát, že se informace zapisují zvlášť do sloupců tabulky, ve skutečnosti se jedná vždy o jednu buňku, ve které jsou jednotlivé položky odděleny mezerami.

Velmi důležitou informací je pozice [X,Y] součástky vzhledem k referenčnímu bodu desky, avšak tyto souřadnice udávají pouze pozici součástky jako celku, neříkají nic o souřadnicích pájecích děr pro osazení součástky. Toto bylo nutné ošetřit programově na úrovni systému Control Web 6, kde byla vytvořena knihovna nejpoužívanějších součástek a v závislosti na pouzdře součástky a její orientaci, byly vytvořenými algoritmy vypočteny přesné souřadnice pájecích děr.

3.2 Přenos dat EAGLE – Control Web

Po úspěšném získání výstupního souboru s potřebnými informacemi o osazených součástkách bylo nutné tyto informace přenést do aplikace v systému Control Web 6. Nejschůdnější variantou jak toto provést, bylo využití DDE komunikace mezi systémy Control Web 6 a Microsoft Excel. Při exportu Partlistu máme na výběr různé formáty výstupního souboru, mezi nimiž je i soubor s příponou *.xls.

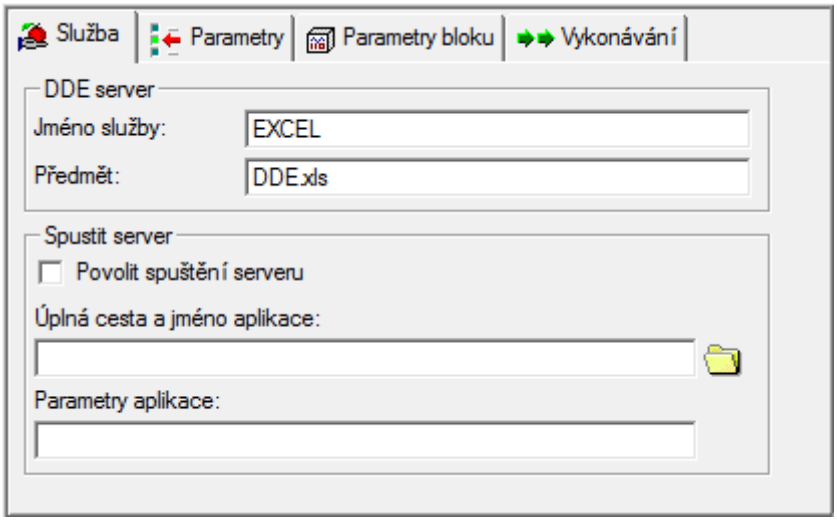
DDE Komunikace

DDE je dynamická výměna dat, sloužící ke sdílení proměnných mezi několika aplikacemi na jednom počítači. Pokud se má sdílet skupina proměnných, tak pro tuto skupinu musí být jeden DDE server a jeden nebo více DDE klientů. DDE server "vlastní" všechny proměnné, klienti se na tyto proměnné odkazují, mohou je číst nebo do nich zapisovat, pokud je toto dovoleno [Dokumentace SCADA systému PROMOTIC].

V této laboratorní úloze je serverem systém Microsoft Excel, respektive exportovaný *.xls soubor s příslušnými daty a klientem aplikace v systému Control Web, která se na tato data odkazuje. Pro zahájení spojení je nutné určit tyto parametry:

- Jméno služby (application) — určuje symbolický název DDE serveru
- Předmět (topic) — určuje téma výměny

Oba tyto důležité parametry se určují v konfiguračním nástroji systému Control Web 6.



Obr. 16 Konfigurace DDE Serveru

Zatímco u serveru, který v tomto případě představuje MS Excel, postačí ke zpřístupnění dat pouze jeho spuštění, respektive otevřený příslušný *.xls soubor, u CW6 klienta je nutné ještě nakonfigurovat tzv. parametrický a mapovací soubor.

Parametrický soubor obsahuje definice kanálů ve specifickém formátu ovladače. Řídí, jaké datové typy kanálů bude ovladač používat ve svých datových strukturách. Struktura souboru je svázána s ovladačem (jedná se o konfigurační soubor ovladače) a parametrické soubory různých ovladačů proto není možné zaměňovat [Moravské přístroje 2010].

Mapovací soubor obsahuje rozsah kanálů, které ovladač může použít. Údaje z tohoto souboru používá překladač, aby ověřil, jestli je typ určený definicí kanálu přípustný. Mapovací soubory se dodávají většinou s ovladači, některé ovladače navíc dokážou tento soubor vytvářet svými pomocnými nástroji [Moravské přístroje 2010].

Příklad zápisu parametrického souboru:

```
[server]                          (*Definice parametrů DDE serveru*)
name = EXCEL
topic = DDE.xls
monitor = FALSE
driver_crash = FALSE
debug = FALSE
run_service = FALSE
connect_attempts = 3
connect_timeout = 100
reconnect_attempts = 20
reconnect_timeout = 200
operation_timeout = 50
operation_attempts = 3
service_path = ""
decimal_char = .
true_text = 1
false_text = 0
item_sep = .
data_sep = 0D0A00
request_limit = 1

[channels]                        (*Definice kanálů DDE serveru*)
1 = string bidirectional, 'R9C1'
2 = string bidirectional, 'R10C1'
3 = string bidirectional, 'R11C1'
4 = string bidirectional, 'R12C1'
```

Příklad zápisu mapovacího souboru:

```
(*Deklarace kanálů DDE serveru*)
begin
      1 -       50 string bidirectional
end.
```

3.3 Přenos dat Control Web – PLC

Pro přenos dat mezi systémem CW6 a PLC byl použit protokol OPC. OLE for Process Control (OPC) představuje první úspěšnou iniciativu standardizující komunikační rozhraní mezi prvky průmyslové automatizace, průmyslovými automaty, čidly a akčními členy na jedné straně a řídicími či operátorskými počítači a průmyslovými informačními systémy na straně druhé. Ze standardu OPC profitují především uživatelé, kteří díky němu přestávají být vázáni na programové a technické vybavení podporující pouze vlastnické protokoly zavedených firem, zejména pokud tyto firmy brání své postavení právními překážkami zabraňujícími implementaci daného protokolu třetím stranám [Moravské přístroje 2010].

Využití OPC přenosu

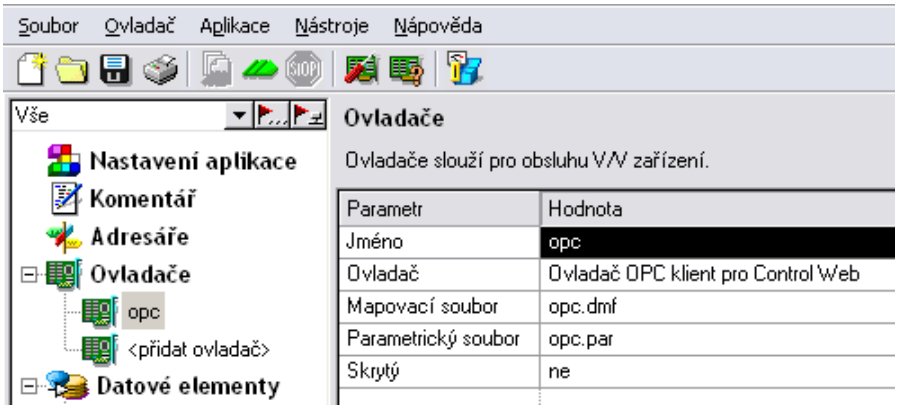
OPC standard podporuje několik způsobů komunikace, aby vyhověl různým požadavkům klientů:

- Synchronní komunikace vždy čekající na přenos dat z/do zařízení.
- Synchronní komunikace pracující s vyrovnávací pamětí serveru.
- Asynchronní komunikace.
- Periodická komunikace serveru se zařízením a zpětné volání klienta při změně dat.

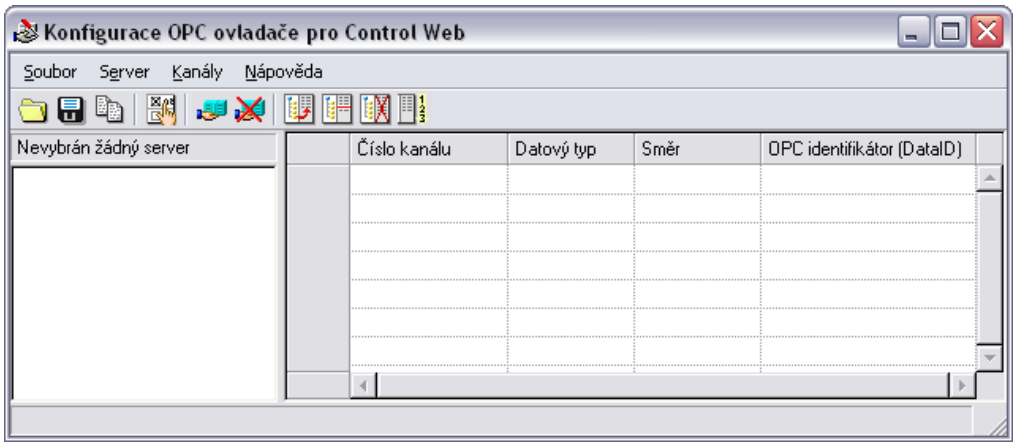
Druhý a čtvrtý způsob komunikace předpokládá, že server sám dokáže vyvolávat komunikaci s periferií a buď jen ukládat přečtená data do vyrovnávací paměti, odkud mohou být synchronně čtena nebo předávat data klientovi zpětným voláním [Moravské přístroje 2010].

Konfigurace OPC komunikace

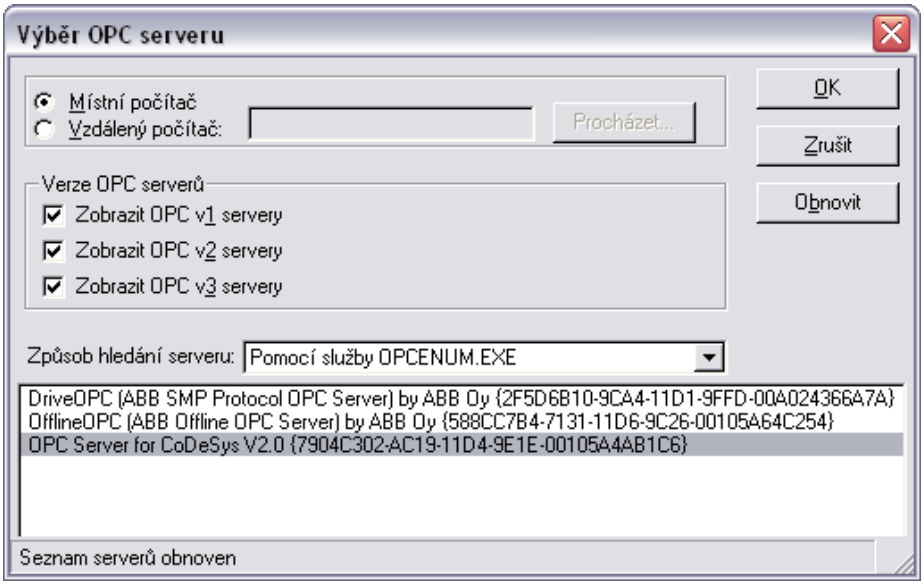
Stejně jako tomu bylo i u DDE komunikace v systému CW6, tak i OPC klient potřebuje dva konfigurační soubory. Parametrický soubor s příponou *.PAR a mapovací soubor s příponou *.DMF. Protože mapovací i parametrický soubor velmi závisí na použitém OPC serveru, je k dispozici nástroj s grafickým rozhraním, umožňující pohodlnou tvorbu obou souborů. Uživatel tak vůbec nepotřebuje znát strukturu těchto souborů, stačí vybrat OPC server, označit položky, které mají být komunikovány jako kanály a o generování konfiguračních souborů se postará program automaticky [Moravské přístroje 2010].



Obr. 17 Ovladač pro OPC Server v systému CW6



Obr. 18 OPC Konfigurátor

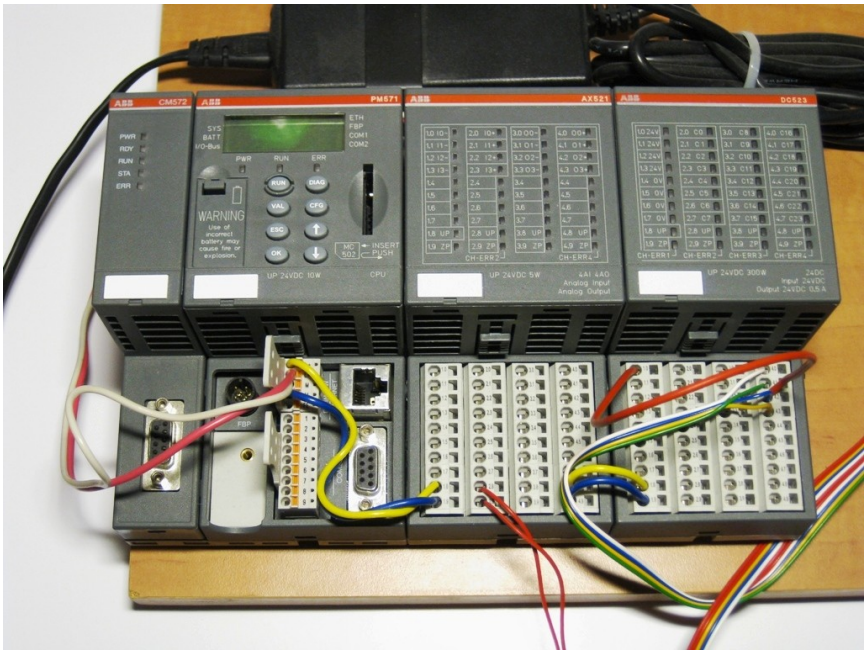


Obr. 19 Výběr OPC Serveru

Okno pro výběr OPC serveru obsahuje jména i CLSID všech OPC serverů instalovaných na lokálním nebo na vzdáleném počítači. Jak je z obrázku dále patrné, na počítači je nainstalován OPC Server pro program CoDeSys, pomocí něhož probíhá komunikace mezi systémem CW6 a PLC, potažmo programem CoDeSys.

3.4 Programovatelný logický automat ABB 500

V této práci bylo použito PLC firmy ABB, typové řady 500. Tato řada má v současnosti již několik nástupců, avšak stále patří k oblíbeným produktům firmy, především díky své jednoduchosti, osvědčené konstrukci a přijatelným pořizovacím nákladům. Díky modulární struktuře, můžeme sestavit mnoho kombinací PLC v závislosti na požadovaném určení. V nabídce firmy je mnoho analogových i digitálních modulů.



Obr. 20 Použité PLC firmy ABB

Pro řízení krokového motoru byl využit zejména digitální modul DC523, jehož digitální výstupy vysílají spínací impulsy pro jednotlivé cívky krokového motoru. Modul obsahuje 24 digitálních pinů, které mohou být nakonfigurovány jako vstupy nebo výstupy. Jejich napěťová úroveň je 0 ÷ 24V.

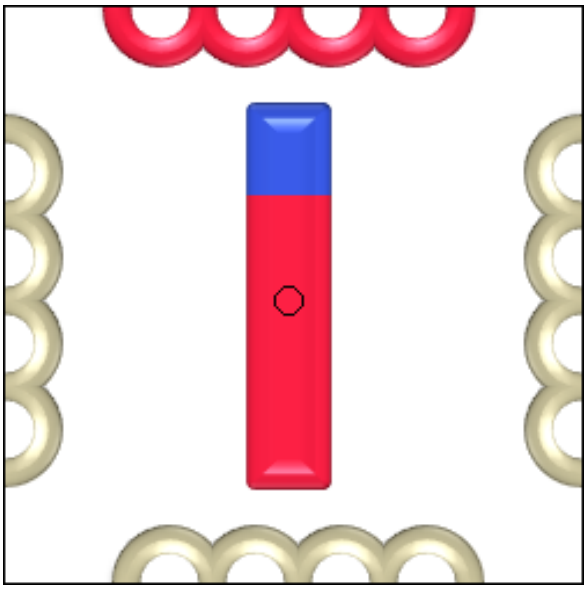
Obr. 21 Digitální modul Modul DC523

3.5 Hardwarová cesta PLC – CNC

Použitý stolní model CNC stroje využívá k polohování frézky krokové motory. Ovládání celého modelu je prováděno pomocí PLC. Nebylo však možné připojit krokové motory přímo na proudové výstupy PLC, protože proud procházející jednotlivými cívkami statorů dosahuje až 4A. Při takovém proudovém zatížení by proudová ochrana PLC provedla okamžitý reset a odstavení, aby nedošlo k poškození elektroniky. Proto bylo nutné navrhnout a vyrobit hardwarový spínací obvod, který by vstupními impulsy spínal silovou část na výstupu.

3.5.1 Princip činnosti krokového motoru

Obecně lze chápat princip činnosti krokového motoru stejně jako u klasického elektromotoru. Krokové motory mají své významné místo v elektrotechnice díky své přesnosti, respektive přesnému pootočení vždy o stejný krok. Jedná otáčka motoru se rovná vždy stejnému počtu pulsů přivedených na cívky elektromotoru. Pokud krokový motor nepřetěžujeme a nedojde k jeho prokluzu, odpadá zde nutnost zpětné vazby měření otáček, protože každou otáčku lze vyjádřit vždy stejným počtem přivedených pulsů. Názorně lze princip krokového motoru vysvětlit následujícím obrázkem.



Obr. 22 Princip činnosti krokového motoru

Pro zjednodušení zde uvažujeme krokový motor se čtyřmi kroky na otáčku. Při průchodu elektrického proudu cívkou statoru dojde k vytvoření nesouhlasného magnetického pole a to k sobě přitáhne rotor. Vhodnou sekvencí spínání a přivádění proudu na jednotlivé cívky dosáhneme otáčení rotoru [Řezáč 2002].

V závislosti na typu použitého krokového motoru existuje několik možností jeho řízení:

Tabulka 1 Typy řízení krokových motorů

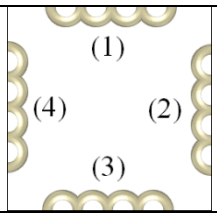
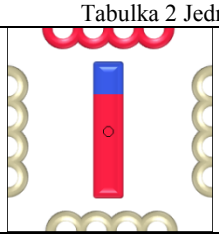
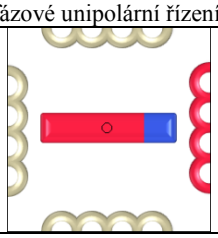
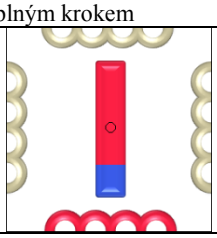
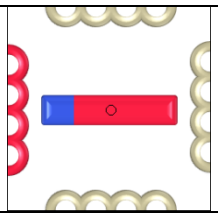
a)	Unipolární řízení	jednofázové	s plným krokem
b)	Unipolární řízení	dvoufázové	s plným krokem
c)	Unipolární řízení	-	s polovičním krokem
d)	Bipolární řízení	jednofázové	s plným krokem
e)	Bipolární řízení	dvoufázové	s plným krokem
f)	Bipolární řízení	-	s polovičním krokem

Pro účely této diplomové práce byly využity pouze varianty a) a b), které budou vysvětleny níže.

Jednofázové unipolární řízení s plným krokem

Při jednofázovém řízení generuje magnetické pole pouze jedna cívka, případně dvojice protilehlých cívek při bipolárním buzení. Řízení s plným krokem znamená, že na jednu otáčku rotoru je zapotřebí přesně tolik pulsů, kolik zubů obsahuje stator. Jednofázové unipolární řízení s plným krokem znázorňuje následující tabulka.

Tabulka 2 Jednofázové unipolární řízení s plným krokem

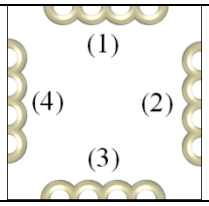
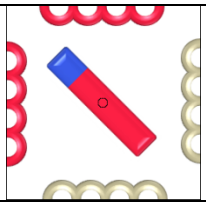
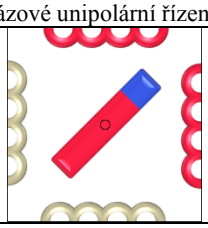
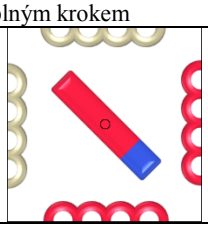
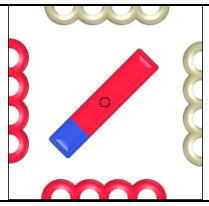
				
Cívka (1)	1	0	0	0
Cívka (2)	0	1	0	0
Cívka (3)	0	0	1	0
Cívka (4)	0	0	0	1

Při unipolárním řízení prochází v jednom okamžiku proud právě jednou cívkou. Motor s tímto buzením má nejmenší proudový odběr, ale také poskytuje nejmenší točivý moment. Výhodou tohoto řešení je jednoduché zapojení řídící elektroniky, kdy v podstatě stačí jeden tranzistor na každou cívku [Řezáč 2002].

Dvoufázové unipolární řízení s plným krokem

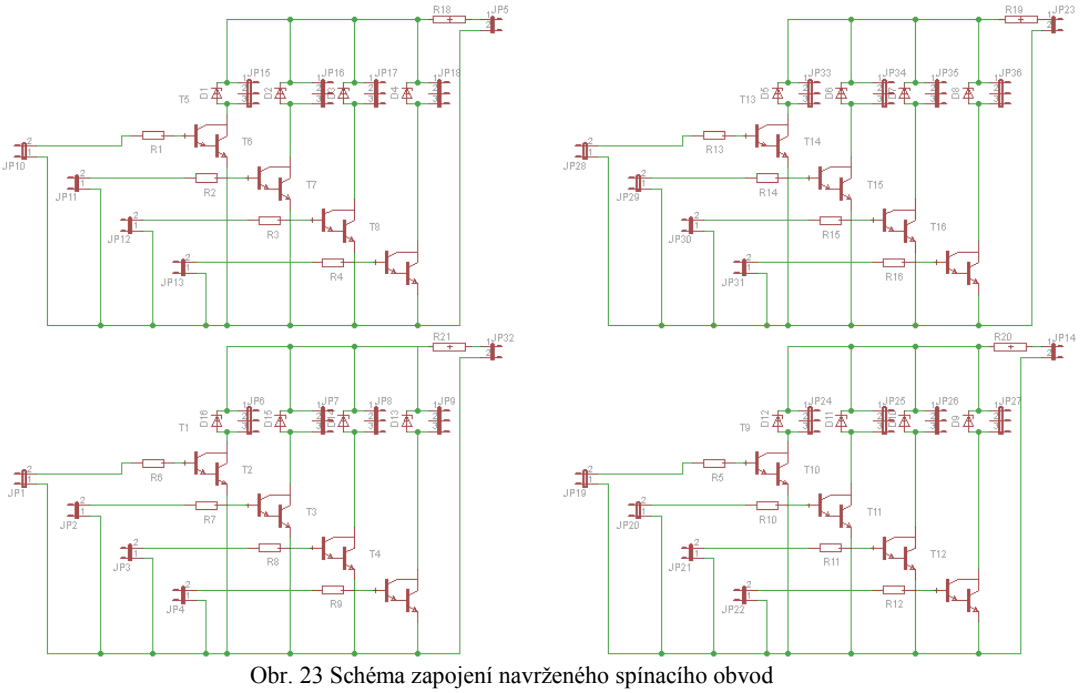
Při dvoufázovém řízení generují shodně orientované magnetické pole vždy dvě sousední cívky. Výhodou tohoto řízení krokového motoru je vyšší točivý moment, nevýhodou dvojnásobná spotřeba energie oproti jednofázovému řízení [Řezáč 2002].

Tabulka 3 Dvoufázové unipolární řízení s plným krokem

				
Cívka (1)	1	1	0	0
Cívka (2)	0	1	1	0
Cívka (3)	0	0	1	1
Cívka (4)	1	0	0	1

3.5.2 Unifikační spínací obvod

Řízení krokových motorů obstarává PLC firmy ABB s využitím diskrétních výstupů digitálního modulu DC523, které však mají maximální proudovou zatížitelnost 0.5A. Tato hodnota je pro přímé řízení motoru nedostačující, protože při plném točivém momentu dosahuje hodnota proudu na každé cívce až 4A. Při takových hodnotách by došlo k okamžitému resetu PLC. Proto bylo nutné navrhnout spínací zesilovací obvod, který by na základě vstupního impulsu z digitálního modulu spínal a přiváděl napětí na jednotlivé cívky. Schéma zapojení navrženého spínacího obvodu je zobrazeno níže, viz. Obr. **23**.

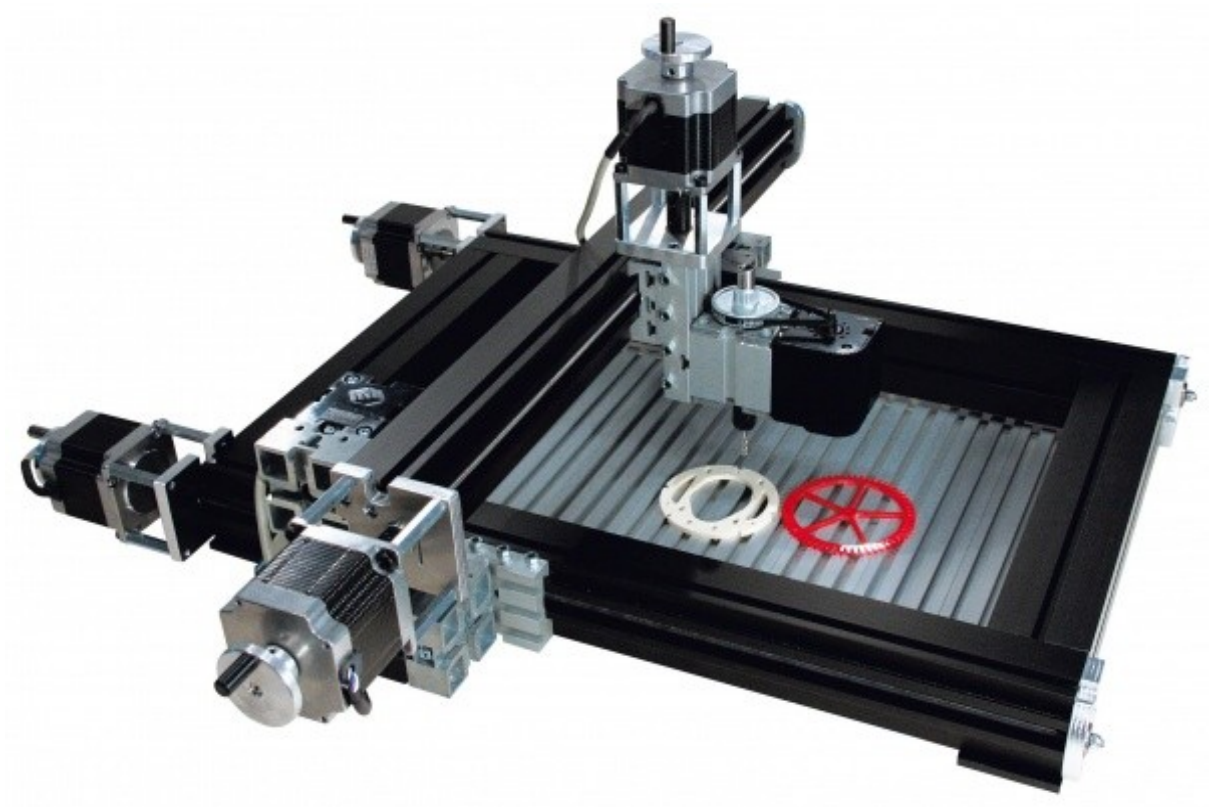


Obr. 23 Schéma zapojení navrženého spínacího obvod

Z uvedeného schématu je patrné, že spínací obvod obsahuje 4 totožné části. Tyto 4 části představují jednotlivé spínací zesilovací obvody pro 4 krokové motory. Vezměme v úvahu část vlevo nahoře. Jednotlivé piny JP10-JP13 představují impulsní vstupy, přivedené z PLC, které otevírají příslušné tranzistory a tím přivádějí napětí na výstupní piny JP15-JP18, určené pro napájení jednotlivých cívek krokového motoru. Přivedením impulsního napětí 24V na vstupní pin dojde k otevření příslušného transistoru (T1 – T4) a tím k přivedení napájecího napětí na příslušný výstupní pin. Vzhledem k tomu, že cívky představují indukční zátěže, je ke každé výstupní části paralelně připojena ochranná dioda.

3.6 Stolní model CNC stroje

Ovládaný model zde představuje celokovová portálová frézka firmy HELAGO.



Obr. 24 Model stolního CNC stroje
[http://www.helago-cz.cz/public/content-images/cz/product/18823.jpg?rand=283816176]

Parametry CNC modelu:

Rozsah pohybu X	0-400mm
Rozsah pohybu Y	0-270mm
Rozsah pohybu Z	0-50mm

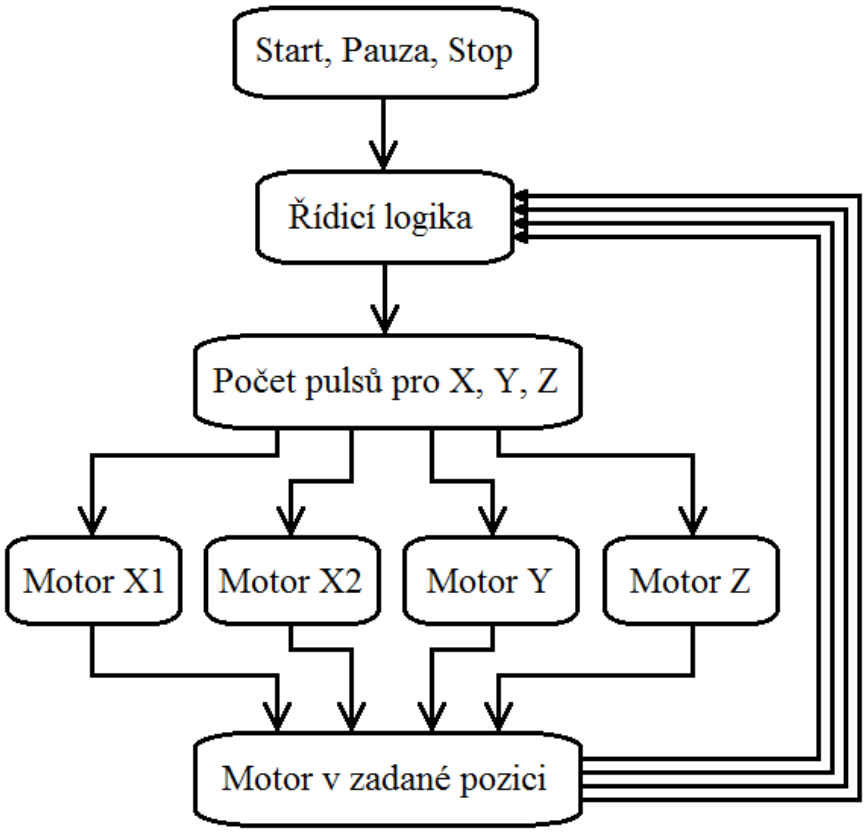
Parametry použitých krokových motorů:

Provozní napětí	24V, SS
Maximální proud	4A
Pootočení za 1 krok	1.8º / krok
Přesnost pootočení / 1 krok	±5% (řízení-celý krok, bez zátěže)
Maximální moment při	200ot/min

Vzhledem k hmotnostem pohyblivých částí modelu CNC a rychlosti digitálního modulu PLC, bylo nutné časování řídicí aplikace nastavit na 5ms. Maximální rychlost krokového motoru je tedy 200 pulsů za vteřinu. To odpovídá jedné otáčce motoru za vteřinu a zároveň při této rychlosti produkuje motor dostatečný točivý moment pro pohybování soustavy bez prokluzu motoru.

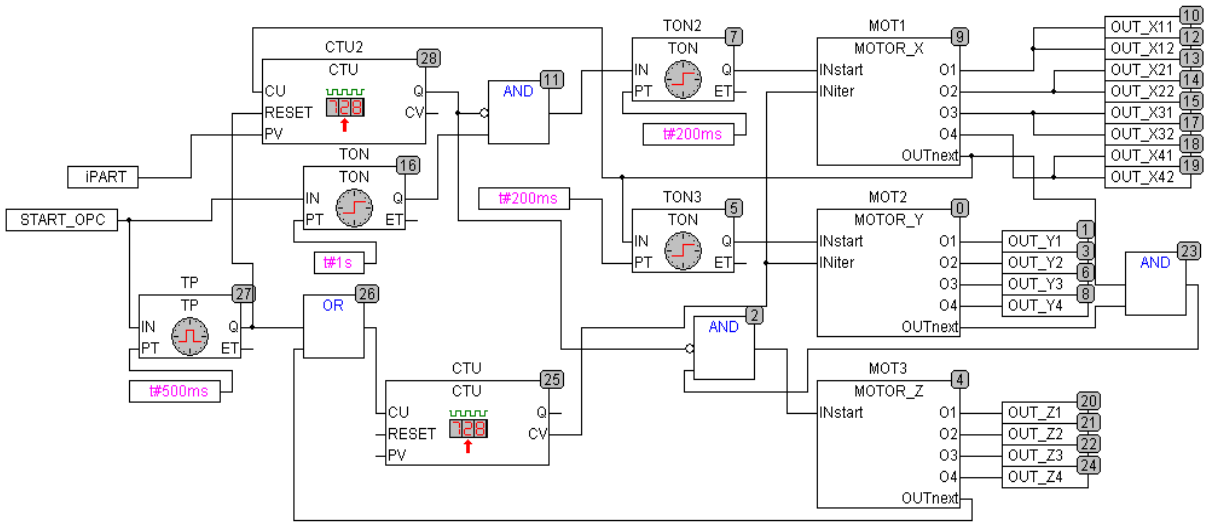
3.7 Ovládací algoritmy pro PLC

Jak již název podkapitoly napovídá, PLC v této úloze figuruje v roli ovládacího prvku, nikoliv však regulačního. Krokové motory zde neobsahují žádnou přímou zpětnou vazbu, která by poskytovala aktuální informace o rotaci motorů, respektive o pohybu CNC v jednotlivých osách. Tyto informace jsou však získávány čítáním pulsů odeslaných na krokové motory, kdy 200 pulsů představuje 1 otáčku motoru. Šroubovice, použité pro pohyb CNC, mají stoupání 1,25 což znamená, že za jednu otáčku motoru se provede lineární posuv o 1,25mm.



Obr. 25 Schéma ovládacího algoritmu pro PLC

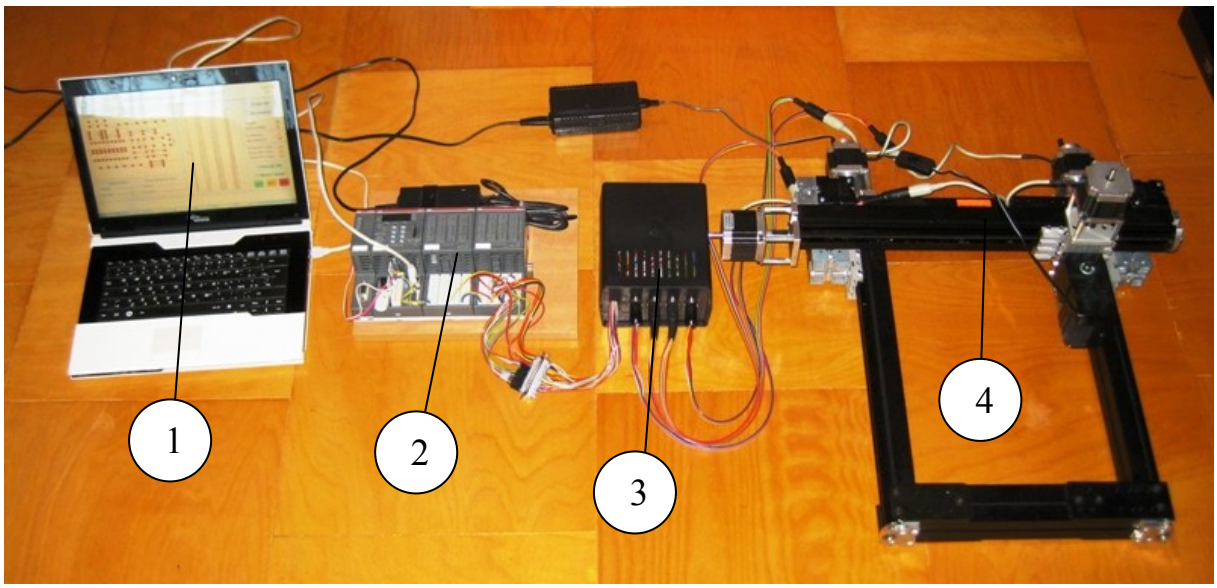
Výše uvedený obrázek představuje hierarchickou a funkční strukturu ovládacího algoritmu pro PLC. Na stejné, nejvyšší úrovni, jsou tři základní funkce programu, spuštění, pauza a zastavení. Tyto signály jsou přijímány prostřednictvím OPC komunikace ze systému Control Web 6 ve formě logických proměnných. Na úrovni řídicí logiky algoritmu bylo nutné zvážit, jak předávat data o jednotlivých pozicích vrtacích bodů desky. Vzhledem k tomu, že na pozadí systému Control Web 6 běží všechny výpočtové algoritmy aplikace, byla i tato data vhodně zpracována a předána systému CoDeSys již ve finální podobě. Na úrovni systému Control Web 6 se tedy všechny metrické vzdálenosti pro pohyb CNC přepočítávají na počet pulsů pro krokové motory, ukládají do polí proměnných a tato pole se prostřednictvím OPC komunikace odesílají do PLC tak, aby PLC před zahájením odvrtávaní již mělo k dispozici všechny nezbytné souřadnice pro ovládání pohybu CNC a nemuselo provádět již žádné výpočtové operace. Řídicí logika ovládacího algoritmu pak jednotlivým blokům pro krokové motory přiděluje jen počty pulsů. Programové bloky pro krokové motory jsou v podstatě nezávislé na celém programu. Mají jeden vstup, přijímající počet pulsů, o který se mají otočit, a jeden výstup, oznamující dokončený rotační pohyb o příslušný počet pulsů. Tyto bloky provedou pohyb motoru vždy, když dojde k odeslání jakéhokoliv počtu pulsů na jejich vstup. Je nezbytné, aby byla správně navržena řídicí logika pro přidělování jednotlivých částek pulsů.



Obr. 26 Část ovládacího programu pro CNC

3.8 Hardwarová kompletace dílčích částí úlohy

Po vytvoření a správném určení funkcí jednotlivých hardwarových částí již bylo možné tyto části propojit do celku, tvořící konečnou hardwarovou podobu laboratorní úlohy.



Obr. 27 Konečná hardwarová podoba laboratorní úlohy

Výše uvedený obrázek znázorňuje propojení dílčích hardwarových částí této úlohy. Na počítači (1) běží vizualizační aplikace, která prostřednictvím OPC protokolu komunikuje s PLC (2). Toto PLC je propojeno se spínacím obvodem (3) jehož digitální vstupy z PLC spínají analogové výstupy pro stolní model CNC stroje (4).

4 Algoritmy pro převod dat z programu EAGLE na souřadnice polohy

Na úrovni systému Control Web bylo nutné z exportovaného *.xls souboru, prostřednictvím DDE komunikace, získat souřadnice jednotlivých součástek vzhledem k referenčnímu bodu [0,0] a v závislosti na konkrétním pouzdře součástky a její orientaci, získat přesné souřadnice [X,Y] vrtacích bodů pro každou součástku. Jak ukazuje Obr. **28**, nevýhodou tohoto způsobu exportu do *.xls souboru je fakt, že důležité informace, se kterými se bude dále pracovat, jako název pouzdra součástky, pozice vzhledem k bodu [0,0] a její orientace, se zapisují do jednoho textového řetězce oddělené mezerami a vždy do jedné buňky tabulky. Bylo tedy nutné vytvořit algoritmus, který by umožňoval tato data separovat a z textového formátu převést na vhodný datový typ.

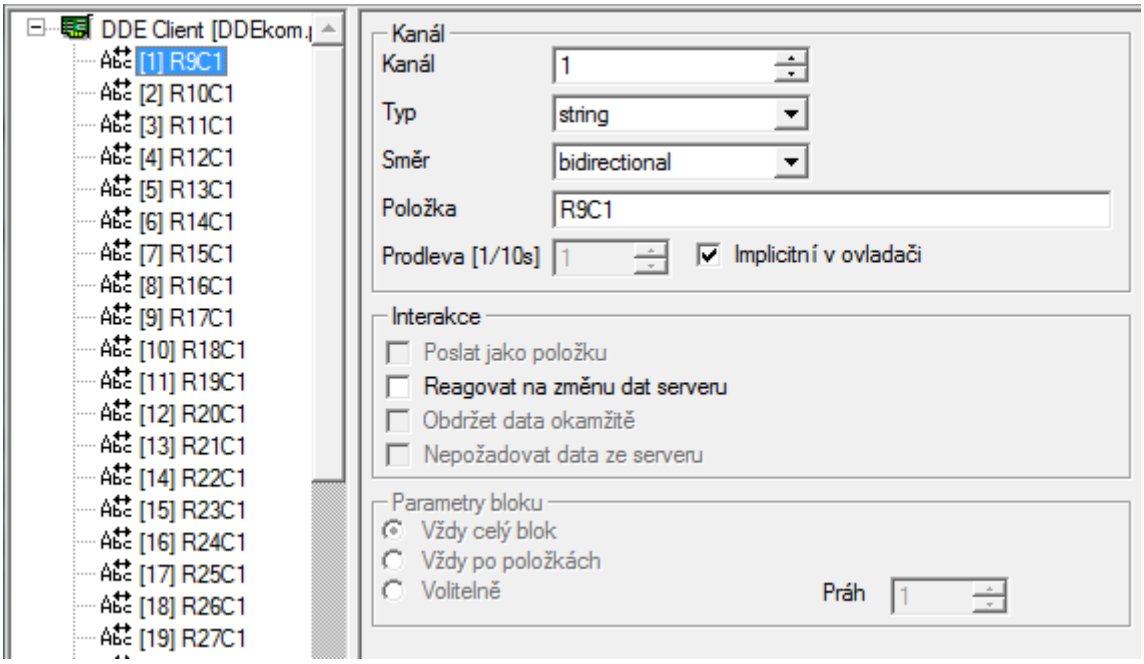
	A	B	C	D	E	F
1	Partlist					
2						
3	Exported from Deska.brd at 22.3.2011 10:42:56					
4						
5	EAGLE Version 5.6.0 Copyright (c) 1988-2009 CadSoft					
6						
7	Part	Value	Package	Library	Position (mm)	Orientation
8						
9	E$1		ARK500/3	GM99	(16.51 6.35)	R180
10	E$2		ARK500/3	GM99	(31.75 6.35)	R180
11	E$3		CK	GM99	(5.08 53.34)	R270
12	E$4		CK	GM99	(10.16 53.34)	R270
13	E$5		CK	GM99	(15.24 53.34)	R270

Obr. 28 Vyexportovaný .XLS soubor z programu EAGLE

4.1 DDE Přenos dat Excel – Control Web

DDE přenos je komunikace typu klient-server, kdy server data poskytuje a klient se na tyto data odkazuje. Serverem je zde vyexportovaný a otevřený *.xls soubor a klientem je aplikace v systému Control Web.

Prvním krok je správná konfigurace DDE serveru a také určení vstupně výstupních kanálů, kterými se budou přenášet vyžádaná data. U těchto kanálů je nutné určit jejich index, směr, datový typ, který budou přenášet a položku viz Obr. **29**.



Obr. 29 Definice DDE kanálů

Položku kanálu zde představuje statické určení konkrétní buňky v *.xls souboru (R9C1 – řada 9, sloupec 1 atp).

Pro tyto účely obsahuje systém Control Web účinný konfigurační nástroj, kde se také tvoří parametrický soubor (*Parametrický soubor obsahuje definice kanálů ve specifickém formátu ovladače. Řídí, jaké datové typy kanálů bude ovladač používat ve svých datových strukturách*.) a mapovací soubor (*Mapovací soubor obsahuje rozsah kanálů, které ovladač může použít. Údaje z tohoto souboru používá překladač, aby ověřil, jestli je typ určený definicí kanálu přípustný*).

Parametrický a mapovací soubor jsou v systému Control Web nezbytné pro vytvoření tzv. ovladače, který řídí datový tok při DDE komunikaci a zároveň zapisuje data z vstupně výstupních kanálů do datových proměnných v systému Control Web.

Parametr	Hodnota
Jméno	DDEkom
Ovladač	Ovladač DDE Client
Mapovací soubor	DDEkom.dmf
Parametrický soubor	DDEkom.par
Mód	run
Skrytý	false

Obr. 30 Vytvoření DDE ovladače

Každý ovladač je spjat s zařízením, se kterým komunikuje. Existují také ovladače, které nepatří k zařízení ale k protokolu a je proto možné je nasadit ke komunikaci s různými zařízeními. Obecně vzato, ovladač vždy komunikuje nějakým protokolem. Zápis a čtení dat z měřicí karty připojené k počítači je rovněž řízen protokolem, i když velmi jednoduchým. Ovladač tedy rozumí svému protokolu a je pomocí něj schopen předávat vstupně/výstupnímu zařízení požadavky aplikace [Moravské přístroje 2010].

Ovladač je standardně pomocí parametrického souboru nastaven na určitou množinu, které určitým způsobem odpovídají datům přímo v technologickém zařízení. Každý vnitřní kanál ovladače má své číslo, které se používá při definici kanálu v aplikaci. Pomocí tohoto čísla tedy dochází k logickému propojení pojmenovaného kanálu, datového elementu, s kanálem. Číslo kanálu v ovladači je základní informace, která se při výměně dat mezi jádrem a ovladačem používá k rozlišení kanálu, a každý požadavek jádra směrovaný k ovladači je proto tímto číslem vybaven [Moravské přístroje 2010].

Kanály

Kanál slouží jako vazba mezi aplikací a V/V zařízením.

Parametr	Hodnota	Popis
name	DDE_STR	Jméno
type	string	Datový typ
low_index	1	Dolní index
high_index	50	Horní index
init_value		Počáteční hodnota
driver	DDEkom	Ovladač
direction	bidirectional	Směr
timeout		Prodleva komunikace
comment		Komentář
color		Barva
mask		Zobrazovací maska pro zobrazení čísel

Obr. 31 Definice V/V kanálů

Posledním krokem pro úspěšnou DDE komunikaci je tedy vytvoření kanálů. Jak zobrazuje Obr. **31**, pro tuto úlohu bylo využito 50 V/V kanálů, přenášejících až 50 textových řetězců, nesoucích informace o daných součástkách plošného spoje. Pro větší množství datových elementů je vhodné využití pole.

4.2 Separace dat z textového řetězce

Po úspěšném přenosu textových řetězců, obsahujících informace o součástkách desky, do aplikace v systému Control Web 6, bylo nutné tyto data separovat a uložit do datových elementů vhodných typů. Vzhledem k vysokému počtu těchto elementů nebylo možné separace provádět individuálně, ale bylo nutné vytvořit obecný algoritmus. Programovací jazyk systému Control Web 6 se svou syntaxí podobá jazyku C. Velkou výhodou a v tomto případě i oporou však byla velké knihovna vnitřních funkcí, která obsahuje i funkce pro práci s textovými řetězci. Formát každého řetězce přijatého z *.xls souboru je následující:

‘E$1 ARK500/3 GM99 (16.51 6.35) R180‘

První položkou je pořadové označení součástky na desce, následuje název pouzdra dané součástky, název knihovny v programu EAGLE, obsahující danou součástku, dále pak souřadnice [X,Y] vzhledem k referenčnímu bodu desky a poslední položkou je natočení součástky ve/proti směru hodinových ručiček o daný úhel.

Pro tyto účely bylo optimální využití vnitřní funkci systému Control Web, item:

```
item( s1, s2 : string; n : longcard ): string
```

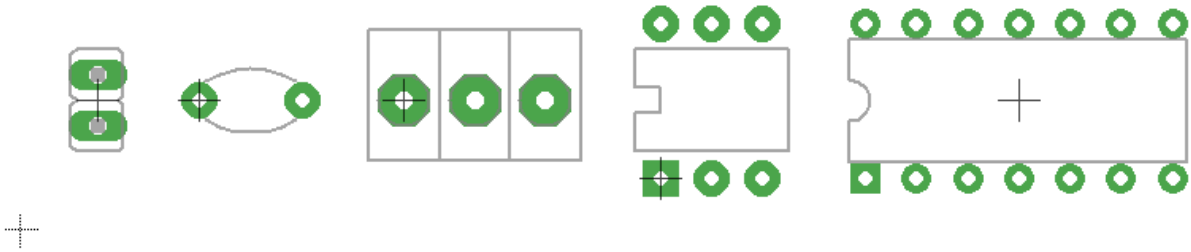
Tato funkce vrací n-tý podřetězec z retězce s1, přičemž jako oddělovače podřetězců používá všechny znaky řetězce s2. Výsledný skript pro separaci řetězců je následující:

```
for iP2 = 1 to ipart do
    DDE_SGN[iP2]=item( DDE[iP2], ' ', 0 );
    DDE_POS[iP2]=item( DDE[iP2], '()', 1 );
    DDE_POSX[iP2]=item( DDE_POS[iP2], ' ', 0 );
    DDE_POSY[iP2]=item( DDE_POS[iP2], ' ', 1 );
    DDE_PACK[iP2]=item( DDE[iP2], ' ', 1 );
    DDE_ORNT[iP2]=item( DDE[iP2], ' ', 5 );
    DDE_POSX_REAL[iP2]=val( DDE_POSX[iP2], 10);
    DDE_POSY_REAL[iP2]=val( DDE_POSY[iP2], 10);
end;
```

Ze základní syntaxe funkce item je zřejmé, že její návratová hodnota je rovněž string řetězec. Proto jsou předposlední dva řádky výše uvedeného skriptu převody z textového datového typu na reálný číselný datový typ. Tyto proměnné představují reálné hodnoty souřadnic [X,Y] referenčního bodu dané součástky.

4.3 Výpočet souřadnic vrtacích bodů

V této části jsou již k dispozici datové proměnné, separované z textových řetězců, obsahující přesné souřadnice [X,Y] vyjadřující vzdálenost jednotlivých součástek vzhledem k referenčnímu bodu desky [0,0]. Každá součástka však má také svůj referenční bod, jehož poloha je právě určena získanými souřadnicemi [X,Y] . Tento referenční bod sice vyjadřuje polohu součástky na desce, bohužel však nic neříká o poloze jednotlivých pájecích otvorů pro součástku. Nevýhodou také je, že každá součástka má tento svůj referenční bod implicitně umístěn jinak, viz. Obr. **32**, a navíc lze jeho polohu editovat.



Obr. 32 Referenční body součástek

Uvedený obrázek znázorňuje několik vybraných součástek s jejich referenčními body. V levém dolním rohu je zobrazen referenční bod desky [0,0] a každá součástka obsahuje také svůj referenční bod, označen křížkem, který určuje polohu [X,Y] vzhledem k bodu [0,0]. Je zřejmé, že algoritmus pro výpočet polohy jednotlivých pájecích otvorů nebylo možné napsat plně dynamicky, protože program EAGLE při exportu sice uvádí polohu součástky na desce, ale neříká nic o poloze referenčního bodu jednotlivé součástky. Z tohoto důvodu jsem algoritmus vytvořil pro konkrétní vybrané součástky a rozdělil jej na statickou a dynamickou část. Statická část výpočtového algoritmu cyklicky testuje název pouzdra součástky a v závislosti na typu součástky předává dynamické části potřebné proměnné pro výpočet souřadnic vrtacích bodů.

Část algoritmu pro patice DIL-08:

```
for iP3=1 to ipart do
…
switch DDE_PACK[iP3] of
…
  case 'DIL-08' ;
      pitch1=3.81;
      pitch2=3.81;
      drl=8;
      goto DRILL4;
…
DRILL4:
      switch DDE_ORNT[iP3] of
      case 'R0' , 'MR0' , 'R180' , 'MR180';
       iDIL=1;
       DRILL_X[idr]=DDE_POSX_REAL[iP3]-pitch1;
       DRILL_Y[idr]=DDE_POSY_REAL[iP3]+pitch2;
       loop
        DRILL_X[idr+1]=DRILL_X[idr]+2.54;
        DRILL_Y[idr+1]=DRILL_Y[idr];
        idr=idr+1;
        if iDIL=(drl/2) then
         exit;
         else
         iDIL=iDIL+1;
        end;
       end;
       iDIL=1;
       DRILL_X[idr]=DDE_POSX_REAL[iP3]-pitch1;
       DRILL_Y[idr]=DDE_POSY_REAL[iP3]-pitch2;
       loop
        DRILL_X[idr+1]=DRILL_X[idr]+2.54;
        DRILL_Y[idr+1]=DRILL_Y[idr];
        idr=idr+1;
         if iDIL=(drl/2) then
          exit;
          else
          iDIL=iDIL+1;
         end;
        end;
       goto FINISH;
…
FINISH:
end;
end;
```

(1) (2) (3)

Část (1) : V této části dochází k cyklickému testování proměnné DDE_PACK[], která obsahuje název pouzdra příslušné součástky. Podle tohoto názvu se pak rozhoduje, jaké proměnné, respektive jejich hodnoty, se předají dále. V tomto případě proměnná pitch1, obsahující horizontální rozteč mezi středem prvního pájecího otvoru zleva a referenčního bodu součástky a proměnná pitch2, obsahující vertikální vzdálenost středů pájecích otvorů patice od referenčního bodu součástky. Proměnná drl obsahuje počet pájecích otvorů dané patice.

Část (2) : V této části algoritmu se testuje proměnná DDE_ORNT[], která obsahuje informaci o natočení součástky o daný úhel, a podle které se pak přepíná mezi příslušnými variantami výpočtu algoritmu.

Část (3) : Poslední část algoritmu je již čistě výpočtová, kde v závislosti na obdržených proměnných se vypočítávají přesné souřadnice pájecích otvorů pro jednotlivé součástky.

Z uvedené části kódu je patrné, že častěji byly použity cykly switch – case , než if – then. To má své opodstatnění v rychlosti vykonávání těchto instrukcí. Výše uvedená část kódu je pouze zlomek z celkového rozsahu algoritmu a při vysokém počtu rozhodujících podmínek dělení algoritmu je výhodnější právě použití přepínače switch – case, jehož vykonání je mnohem rychlejší než cykly if – then.

Získané hodnoty jednotlivých souřadnic se pak automaticky zapisují do datových proměnných, kde se s nimi bude dále pracovat.

Tabulka 4 Souřadnice děr pájecích otvorů

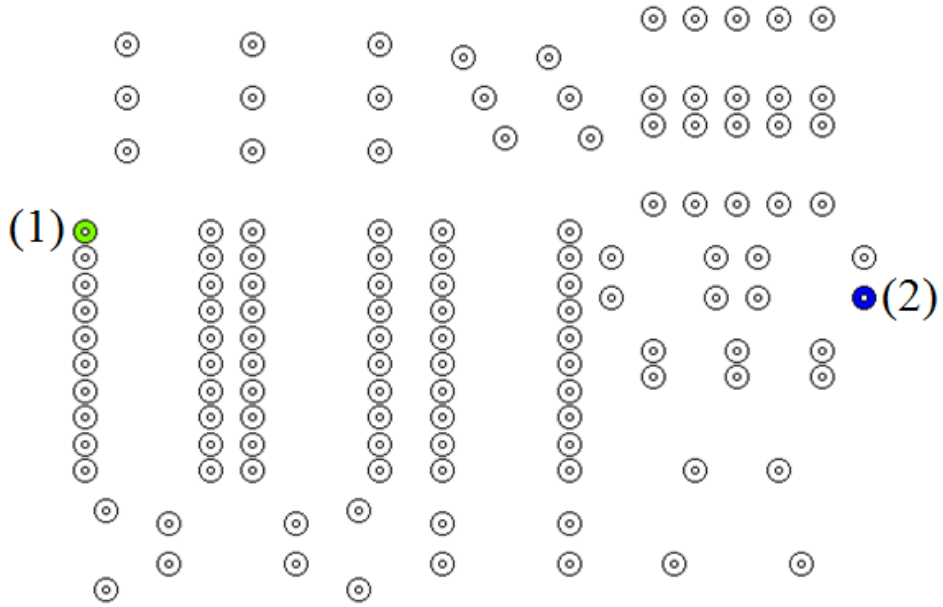
Poloha X	Poloha Y	Soucastka
16.51	6.35	E$1
11.43	6.35	E$1
6.35	6.35	E$1
31.75	6.35	E$2
26.67	6.35	E$2
21.59	6.35	E$2
5.08	53.34	E$3
5.08	48.26	E$3
10.16	53.34	E$4
10.16	48.26	E$4
15.24	53.34	E$5

4.4 Referenční přepočet souřadnic vrtacích bodů

Výše uvedenými algoritmy jsme získali přesné souřadnice vrtacích bodů dané desky pro zvolené součástky. Tyto souřadnice však určují polohu vrtacích bodů pouze v počítači, zatím však nijak nesouvisí s reálnou deskou na CNC stroji. Je zřejmé, že by bylo velmi pracné a náchylné na chybu nastavovat danou fyzickou desku plošných spojů vůči frézce CNC stroje. Naopak mnohem přesnější a spolehlivější je přizpůsobit jednotlivé souřadnice vrtacích bodů dané desce na CNC stroji. Tento proces se skládá z několika dílčích částí:

1) Referenční body desky

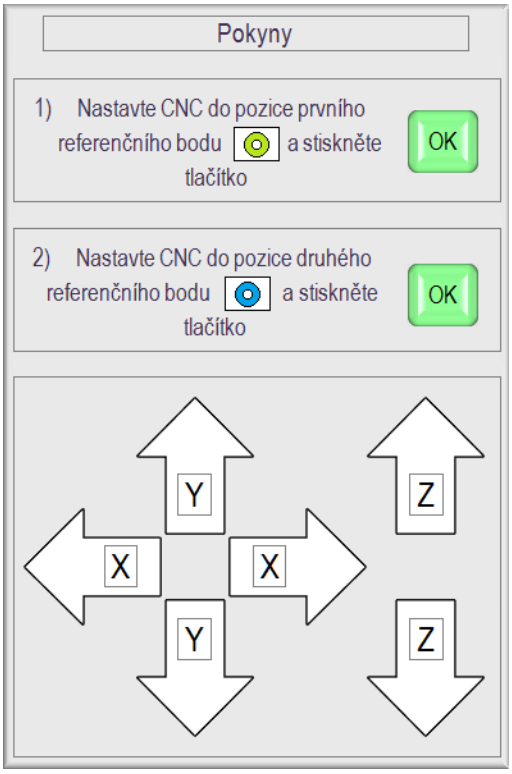
Prvním krokem přizpůsobení souřadnic dané desce bylo určení dvou základních referenčních bodů. Algoritmus pro jejich výběr je navržen tak, že ze všech bodů desky je vybrán první bod nejprve nejvíce vlevo a pak nejvíce nahoře, druhý bod nejprve nejvíce vpravo a pak nejvíce dole. Tyto dva body určují tzv. referenční vektor.



Obr. 33 Referenční body desky plošných spojů

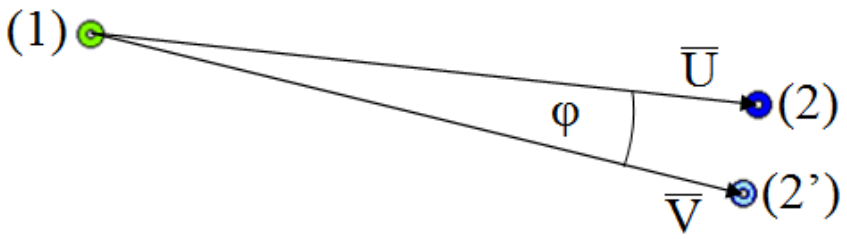
2) Výpočet úhlu pootočení

Jakmile jsme získali potřebné referenční body desky, můžeme je nyní „přenést“ na fyzickou desku. Pomocí panelu pro manuální ovládání CNC modelu (Obr. **34**) nastavíme frézu přesně na pozici prvního referenčního bodu desky, tím fyzicky nastavíme počátek pro další polohování modelu. Poté je nutné nastavit frézu co nejpřesněji na pozici druhého referenčního bodu. Při této činnosti dochází na pozadí aplikace v Control Web 6 k zaznamenávání pohybu frézy a výpočtu souřadnic nově vzniklého bodu.



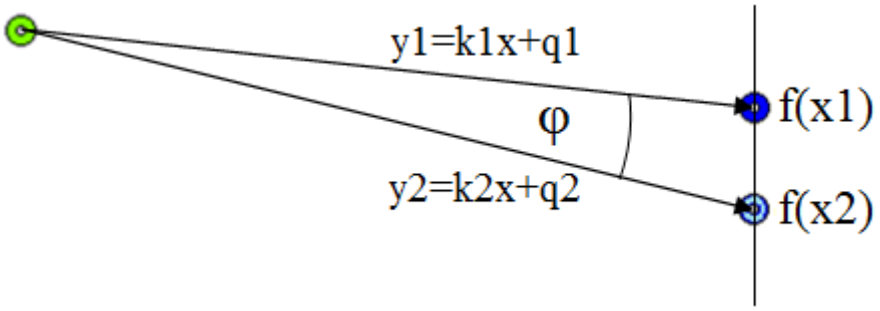
Obr. 34 Manuální ovládání CNC modelu

Přesnost nastavení obou referenčních bodů je rozhodující. Zatímco od prvního z nich se dopočítávají souřadnice ostatních bodů, druhý referenční bod určuje pootočení všech bodů o určitý úhel kolem tohoto bodu.



Obr. 35 Referenční vektory

Výše uvedené vysvětluje Obr. **35**. Vektor U je dán staticky výpočtem. Naopak vektor V jsme získali zaznamenáváním pohybu frézy do pozice bodu (2). Mezi těmito vektory vznikne úhel, který vyjadřuje pootočení všech ostatních bodů. Nyní jsme získali velikost úhlu otočení všech bodů kolem referenčního bodu. Je však nutné znát i orientaci, respektive to, jestli je třeba body otočit ve/proti směru hodinových ručiček. Zjednodušeně způsob rozhodování znázorňuje Obr. **36**.



Obr. 36 Referenční přímky

Pro určení směru otáčení všech bodů je nutné z referenčních vektorů vytvořit tzv. referenční přímky o rovnicích ve směrovém tvaru y=kx+q. Po vytvoření těchto rovnic se provádí porovnávání funkčních hodnot **y** obou přímek pro x-ovou souřadnici referenčního bodu (2). Je zřejmé, že pokud se všechny body otáčejí kolem jednoho referenčního bodu, bude každý bod při svém pohybu opisovat kružnici. Při porovnávání funkčních hodnot obou přímek se dosazuje x-ová souřadnice referenčního bodu (2), souřadnice referenčního bodu (2‘) se zanedbává. Rozhodovací logika je pak následující:

f(x1) < f(x2) … Otočení proti směru hodinových ručiček

f(x1) > f(x2) … Otočení ve směru hodinových ručiček

Nutnou podmínkou pro toto vyhodnocení a celkově pro nastavování referenční polohy fyzické desky je to, aby fyzická deska plošných spojů na CNC stroji byla umístěna **rovnoběžně se směrem pohybu X.**

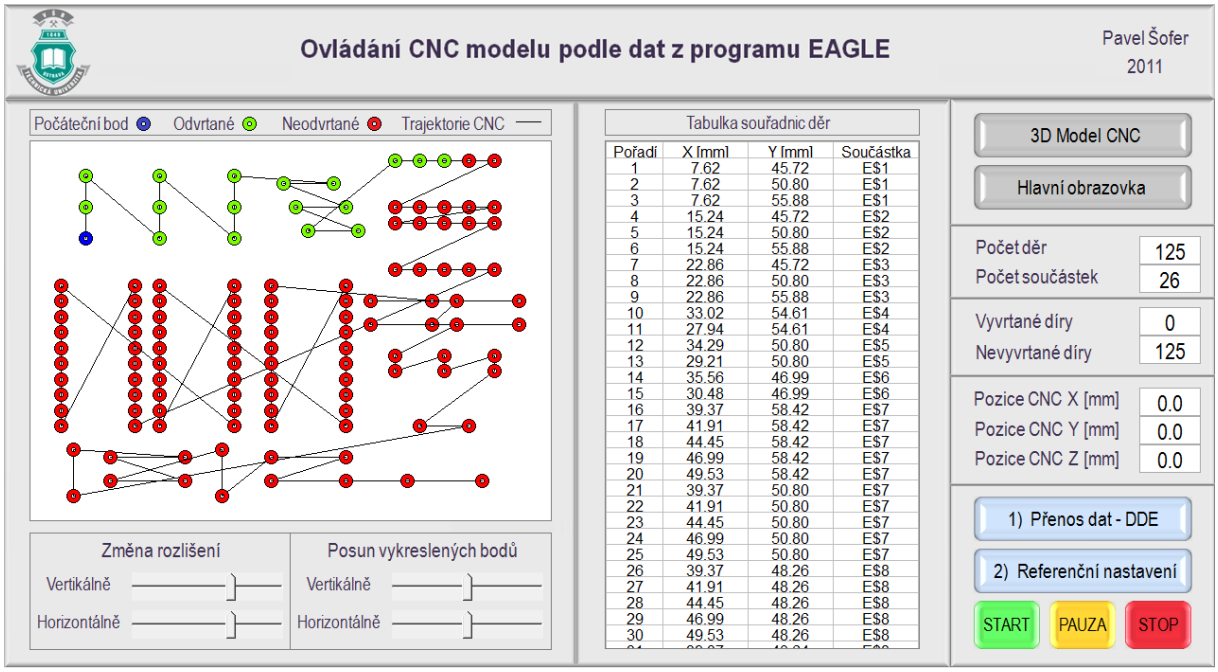
3) Výpočet úhlu pootočení

V této fázi jsou již známy potřebné referenční body, úhel pootočení ostatních bodů i směr tohoto pootočení. Cyklickými algoritmy se již pouze provede přepočet souřadnic [X,Y] všech ostatních bodů. Referenční bod (1) je nyní brán jako počátek a má novou souřadnici [0,0].

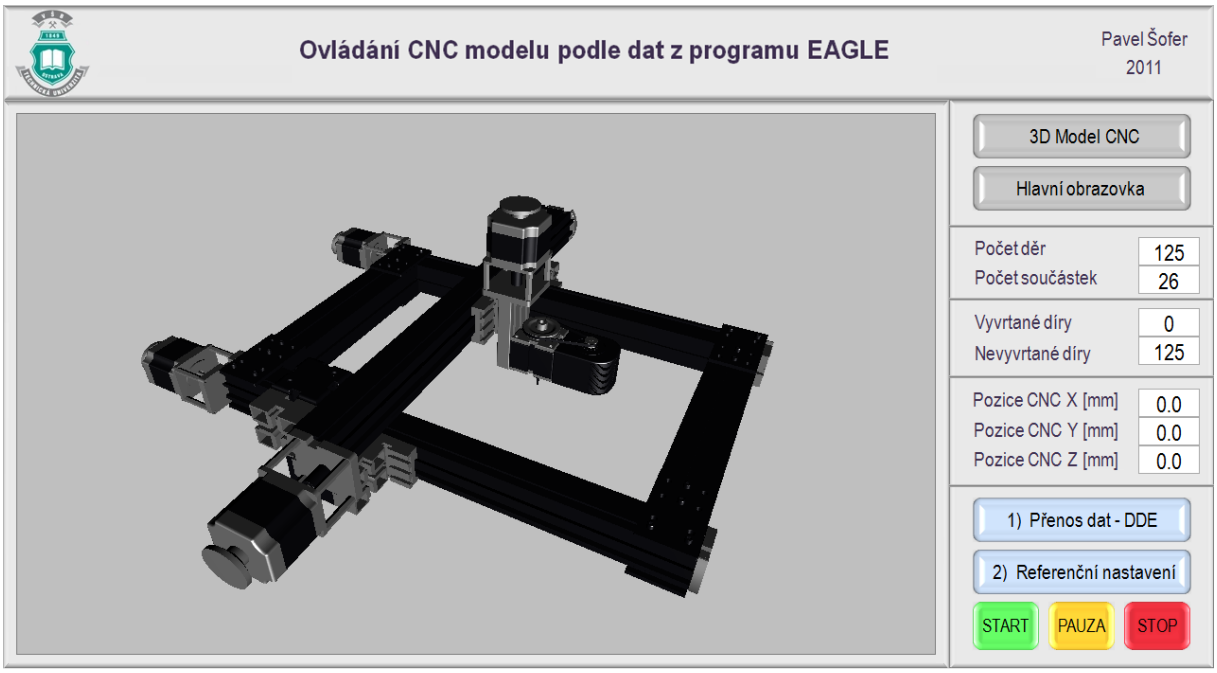
5 Vizualizační aplikace v Control Web 6

Z pohledu celé laboratorní úlohy je důležitá nejen správná činnost výpočtových procesů na pozadí aplikace v Control Web 6, ale také grafické znázornění celé úlohy tak, aby měl uživatel přesné informace o tom, co se právě děje. Za tímto účelem byla vytvořena vizualizační aplikace v systému Control Web 6, která toto umožňuje. Hlavními částmi této aplikace jsou:

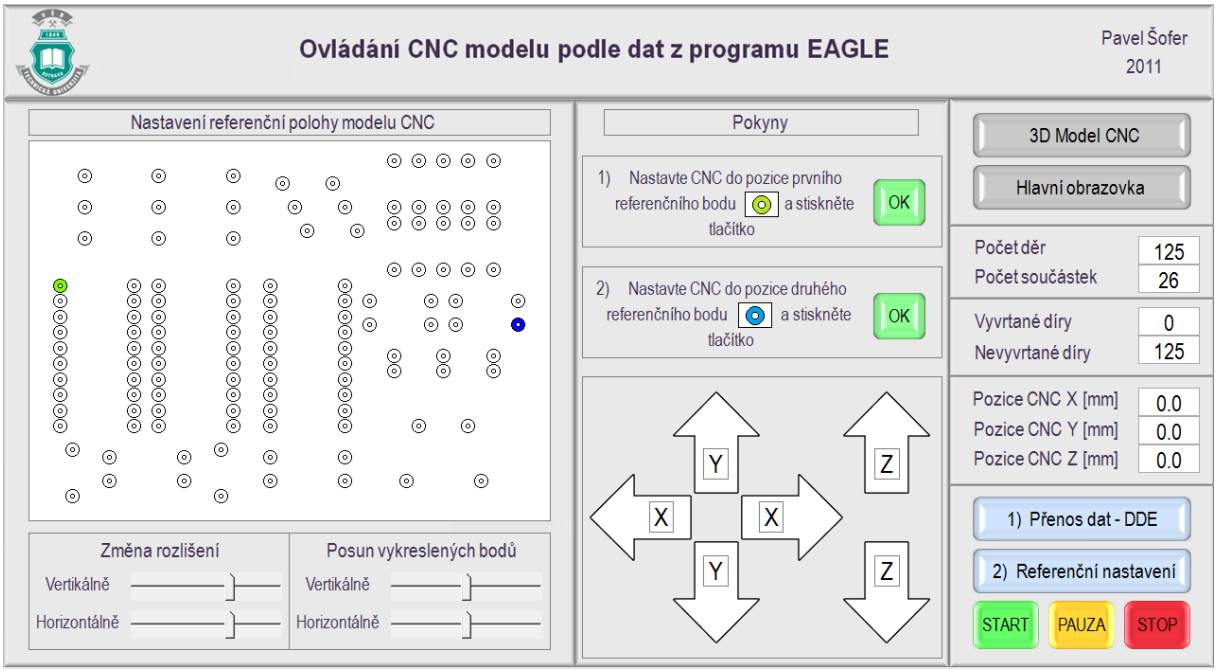
- Grafika desky plošného spoje
- Animační 3D model
- Ovládací a přepínací panel
- Tabulkové informace o desce plošného spoje
- Aktuální informace o vrtacím procesu
- Obrazovka pro nastavení referenční polohy CNC modelu



Obr. 37 Vizualizační aplikace v Control Web 6 – Hlavní obrazovka



Obr. 38 Vizualizační aplikace v Control Web 6 – Animační 3D model

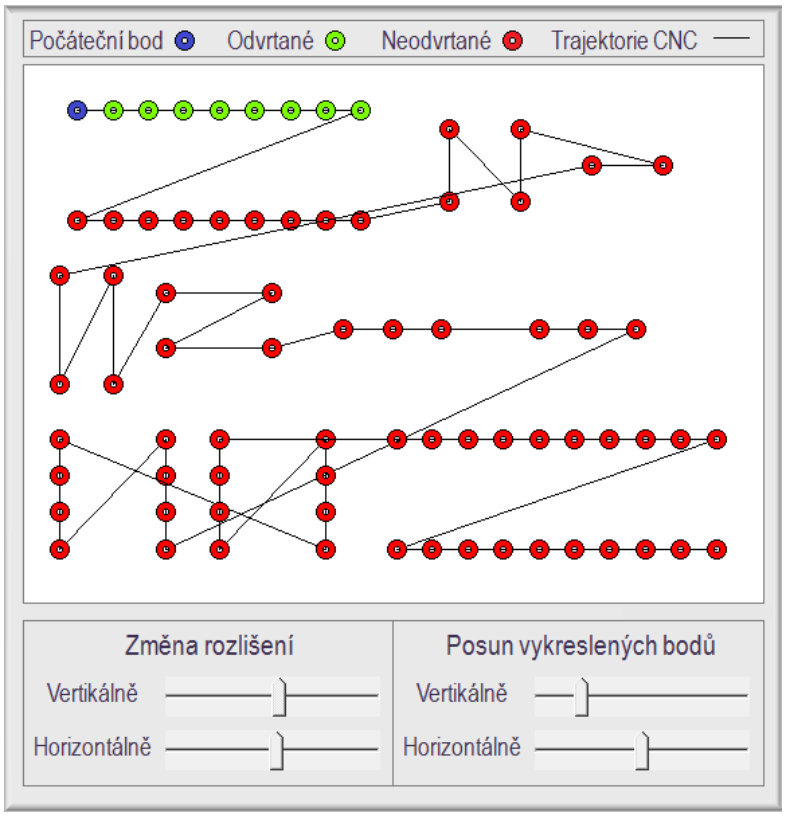


Obr. 39 Vizualizační aplikace v Control Web 6 – Nastavení referenční polohy

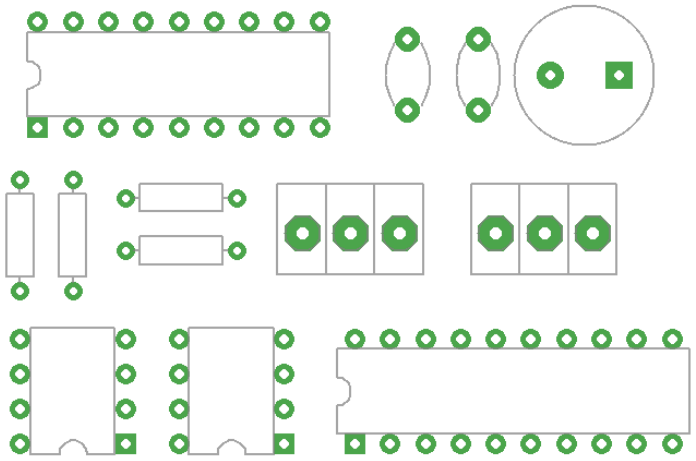
5.1 Grafické zobrazení desky plošných spojů

Grafické znázornění navržené desky plošných spojů zobrazuje Obr. **40**. Pro srovnání a důkaz o správné funkčnosti algoritmů na převod a vykreslení je na Obr. **41** zobrazena tatáž deska z prostředí programu EAGLE.

Toto grafické zobrazení umožňuje vykreslit přesný poziční duplikát děr navržené desky, automaticky vybere a označí díru, která se bude odvrtávat jako první, označí odvrtané a neodvrtané díry a zároveň vybere a vykreslí pořadí vrtání, respektive trajektorii pohybu CNC stroje.



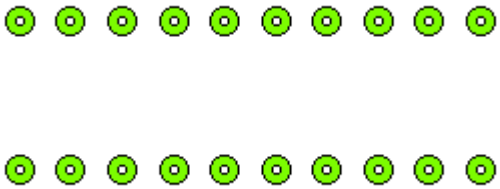
Obr. 40 Grafické zobrazení desky plošných spojů



Obr. 41 Deska plošných spojů v programu EAGLE

Využití přístroje Draw

Pro vykreslování pájecích otvorů podle vypočtených souřadnic byl použit přístroj Draw, obsažen v paletě přístrojů systému Control Web 6. Přístroj Draw slouží pro zobrazení vektorové kresby složené ze základních geometrických prvků (přímek, elips, obdélníků apod.). Rozměry jednotlivých prvků mohou být svázány s výrazy a mohou se tedy s časem měnit. Přístroj Draw dokáže animovat zobrazení, která není možno realizovat standardními přístroji [Moravské přístroje 2006]. Díky přístroji Draw tedy můžeme vytvářen různé obrazce složené z elementárních tvarů.



Obr. 42 Přístroj Draw – Vykreslení patice DIL-20

Vytvoření každého obrazce je dáno několika kroky:

- Vytvoření elementárního objektu
- Nastavení základních vlastností objektu (barva výplně, barva obrysu…)
- Manipulace s objektem (poloha, velikost, napojení proměnných…)

Následující program popisuje vytvoření obrazu desky plošného spoje, viz. Obr. **40**.

```
procedure OnActivate();
label
S1,S2,S3;
begin
for iDraw=1 to dr_count do
   if vypocet=false then
      for iDr=1 to dr_count do
      drawDRILL_X[iDr]=draw_shiftX+(draw_scaleX*DRILL_X[iDr]);
      drawDRILL_Y[iDr]=draw_shiftY+(draw_scaleY*DRILL_Y[iDr]);
      end;
      vypocet=true;
      drawDRILL_X[dr_count+1]=drawDRILL_X[dr_count];
      drawDRILL_Y[dr_count+1]=drawDRILL_Y[dr_count];
      EncodeRGB( 124,252,0, zelena );
      EncodeRGB( 255,255,255, bila );
      EncodeRGB( 0,0,255, modra );
      EncodeRGB( 255,0,0, cervena );
      iDr=1
   elsif vypocet=true then
   goto S1;
   end;
```

(1)

```
S1:
   if state=false then
   goto S2;
   elsif state=true then
   goto S3;
   end;
S2:
   ObjectCreate( otBox, 0, BOX[1] );
   ObjectSetFillMode( BOX[1], fmOutlineFill );
   ObjectSetSolidPaper( BOX[1], bila);
   ObjectSetInkColor( BOX[1], bila);
     for iDr=1 to dr_count do
     ObjectCreate( otLine, 0, LINE[iDr] );
     ObjectCreate( otRing, 0, RNG[iDr] );
     ObjectSetFillMode( RNG[iDr], fmOutlineFill );
     end;
   state=true;
S3:
   ObjectInvalidateRect( BOX[1] );
   BoxSetParameters( BOX[1],0, 0,2000,1000);
   ObjectInvalidateRect( RNG[iDraw]);
   RingSetParameters( RNG[iDraw],drawDRILL_X[iDraw],
                              drawDRILL_Y[iDraw],7,7,2,2);
   ObjectInvalidateRect( LINE[iDraw]);
   LineSetParameters( LINE[iDraw],drawDRILL_X[iDraw],drawDRILL_Y[iDraw],
                         drawDRILL_X[iDraw+1],drawDRILL_Y[iDraw+1]);
end;
end_procedure;
```

(2)

(3)

Část (1) : V této části probíhá počáteční výpočet proměnných polohy jednotlivých vrtacích otvorů. Rovněž je také nutné definovat proměnnou, která bude vyjadřovat barvu pro výplň/obrys jednotlivých objektů. K tomu slouží procedura EncodeRGB, která vytvoří barvu, složenou ze tří základních složek barev (červená, zelená, modrá).

Část (2) : Hned na začátku algoritmu je zřejmé, že celý algoritmus se provádí cyklicky od 1 do proměnné *dr_count*, která se určuje při výpočtu souřadnic děr a udává celkový počet děr desky. V druhé části kódu dochází k vytváření jednotlivých objektů, ze kterých je celkový obraz složen. To znamená, že se vytvoří celkem *dr_count* malých kruhů a *dr_count-1* čar, které tvoří spojnice mezi jednotlivými kruhy a vyjadřují trajektorii pohybu CNC. Objekty se vytvářejí procedurou ObjectCreate a jejich vlastnosti se určují procedurou ObjectSet… .

Část (3) : V poslední části algoritmu dochází k cyklickému překreslování jednotlivých objektů při jejich pohybu. Pokud totiž chceme s jakýmkoliv objektem pohybovat, musíme nejdříve oblast kde je tento objekt vykreslen zneplatnit, překreslit a následně tento objekt vykreslit na jiné, námi žádané pozici. K tomu slouží procedura ObjectInvalidateRect, která zneplatní oblast uvnitř obdélníku opsaného kolem daného objektu.

5.2 Animační 3D model

Součástí vizualizační aplikace je i animační 3D model, zobrazující pohyb CNC modelu v reálném čase. Při výběru jednotlivých softwarů pro tvorbu 3D modelu byl hlavní důraz kladen na podporu importních/exportních 3D formátů, které mohou být dále zpracovány na 3D model ve formátu, vhodném pro systém Control Web 6.

Pro vytvoření 3D Modelu jsem zvolil software Autodesk Inventor. Systém Inventor umožňuje velmi intuitivní tvorbu 3D modelu, jehož konečnou podobu představuje sestava, složená z elementárních částí modelu. Každou část modelu je však nutné namodelovat odděleně. Zjednodušeně lze říci, že modelování součásti začíná jejím 2D náčrtem a následně vysunutím tohoto náčrtu do prostoru.

Obr. 43 Schéma tvorby animačního 3D modelu

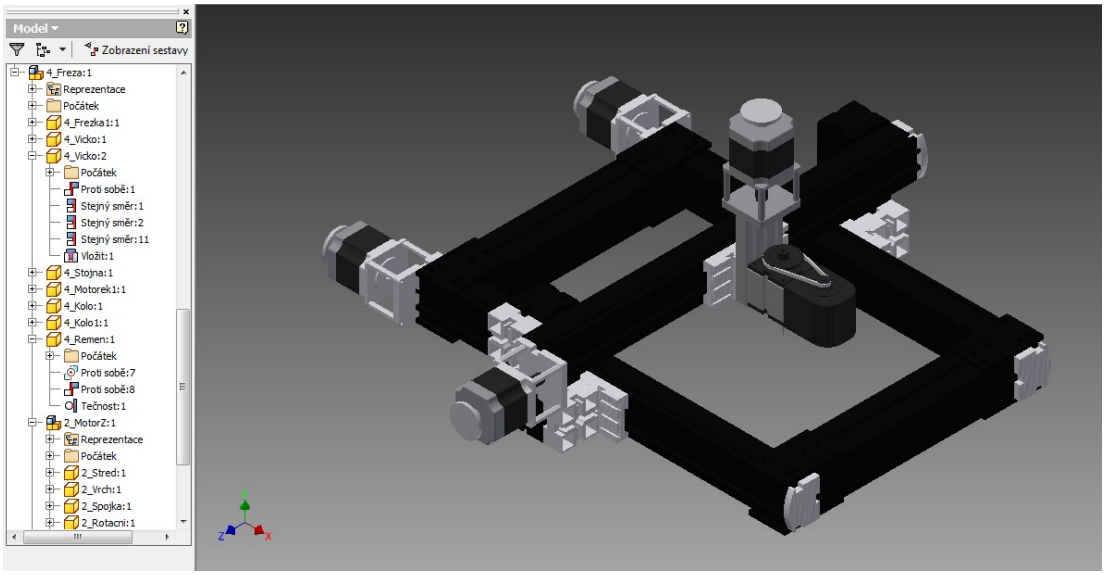
Důležitým mezistupněm mezi konstrukčním softwarem Autodesk Inventor a vizualizačním softwarem Control Web 6 je program, který umí zpracovat vytvořenou sestavu ze systému Inventor a zpracovat ji na 3D model, vhodný pro systém Control Web 6. Programů, které umí tyto sestavy zpracovávat a převádět existuje několik, zvolil jsem však systém Autodesk 3DS Max, jehož pracovní prostředí je velmi přehledné a efektivní, navíc vzhledem k tomu, že se jedná o produkt stejného výrobce jako systém Inventor, zaručuje i výbornou kompatibilitu s tímto systémem.

Autodesk Inventor

Prvním krokem pro vytvoření animačního modelu je pokud možno co nejpřesnější namodelování CNC stroje v programu Autodesk Inventor.

Program Inventor je konstrukčním nástrojem pro tvorbu 3D modelů a 2D výkresové dokumentace. Lze pomoci něj tvořit i prezentace, fotorealistické vizualizace a animace, rovněž správu dokumentů a konstrukčních dat. Základ konstruování v Inventoru tvoří součásti, jejichž geometrie může být odvozena od parametrických 2D náčrtů. Tyto součásti pak mohou být kombinovány a vázány různými typy vazeb do sestav. Při změně kóty, parametru nebo geometrie je automaticky regenerována a aktualizována celá 3D sestava, včetně její výkresové dokumentace.

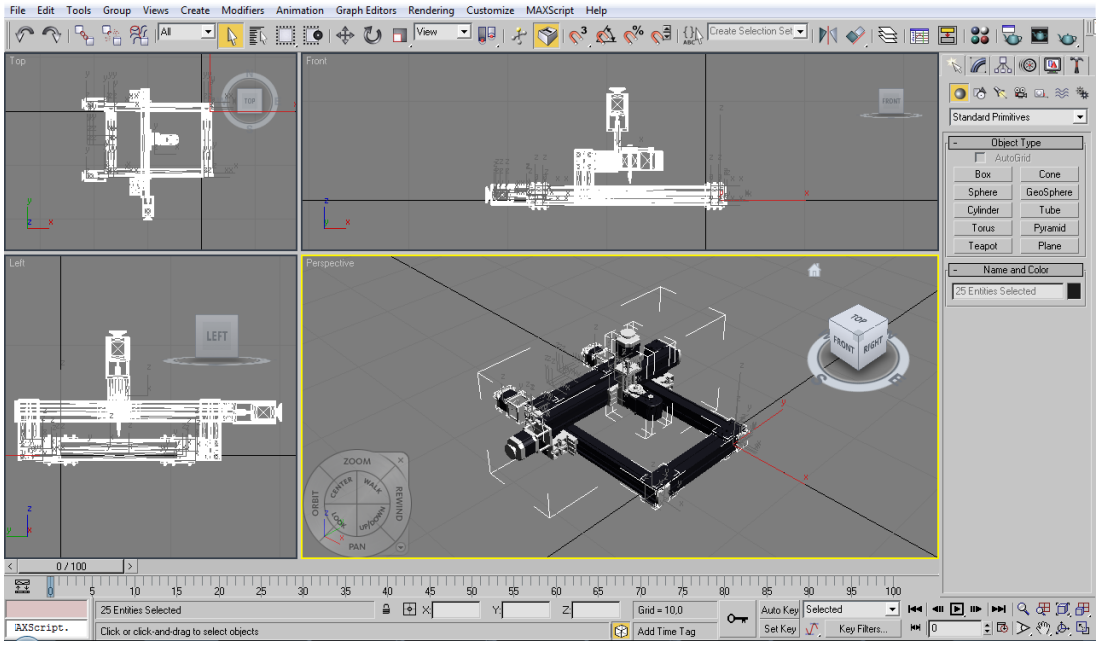
Vytvoření animačního modelu tedy začíná v softwaru Autodesk Inventor, kde je nutné pokud možno co nejpřesněji namodelovat jednotlivé součásti, které tvoří CNC model. Tyto součásti se modelují každá zvlášť do oddělených souborů a ty se pak vkládají do sestavy, ve které je nutné nadefinovat všechny potřebné vazby. Hotová sestava se pak uloží do souboru sestavy, se kterým se bude dále pracovat v programu Autodesk 3DS Max.



Obr. 44 Vytvořený 3D model - Autodesk Inventor

Autodesk 3DS Max

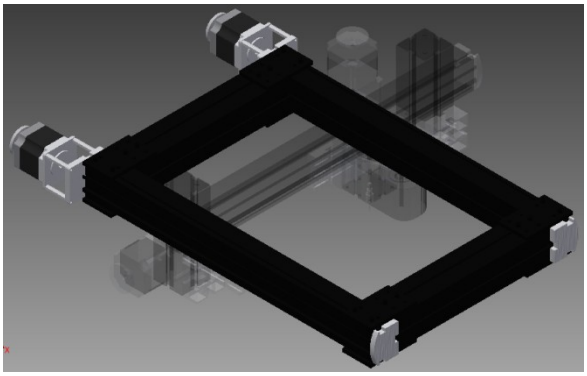
Program Autodesk 3DS Max je animační software, umožňující import vytvořeného konstrukčního modelu, vytvoření animačních vazeb a jeho následný export do formátu 3DS, vhodného pro systém Control Web 6.



Obr. 45 Importovaný model - Autodesk 3DS Max

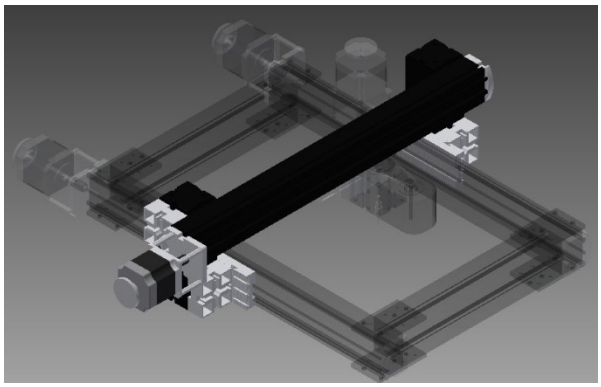
Pro vytvoření animačního modelu CNC stroje v prostředí systému 3DS Max je nutné model načíst po skupinách zajišťujících požadovanou hybnost animace modelu v Control Webu. Každá část modelu se po importu do programu 3DS Max musí pojmenovat. Vhodné je také každé vložené skupině přiřadit barevnou hladinu pro sestavování do bloků [Matušek 2006]. Žádanou animaci CNC modelu zajistíme rozdělením modelu na části, které mají vykonávat nezávislý pohyb v rámci celkového modelu:

- Základna modelu



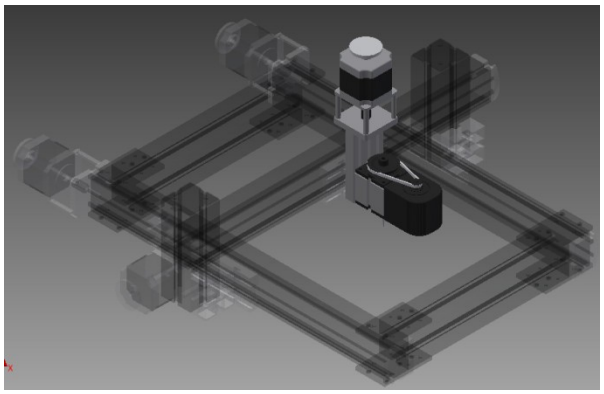
Obr. 46 3D Model - Základna modelu

- Rám s frézkou, posuvný ve směru osy X



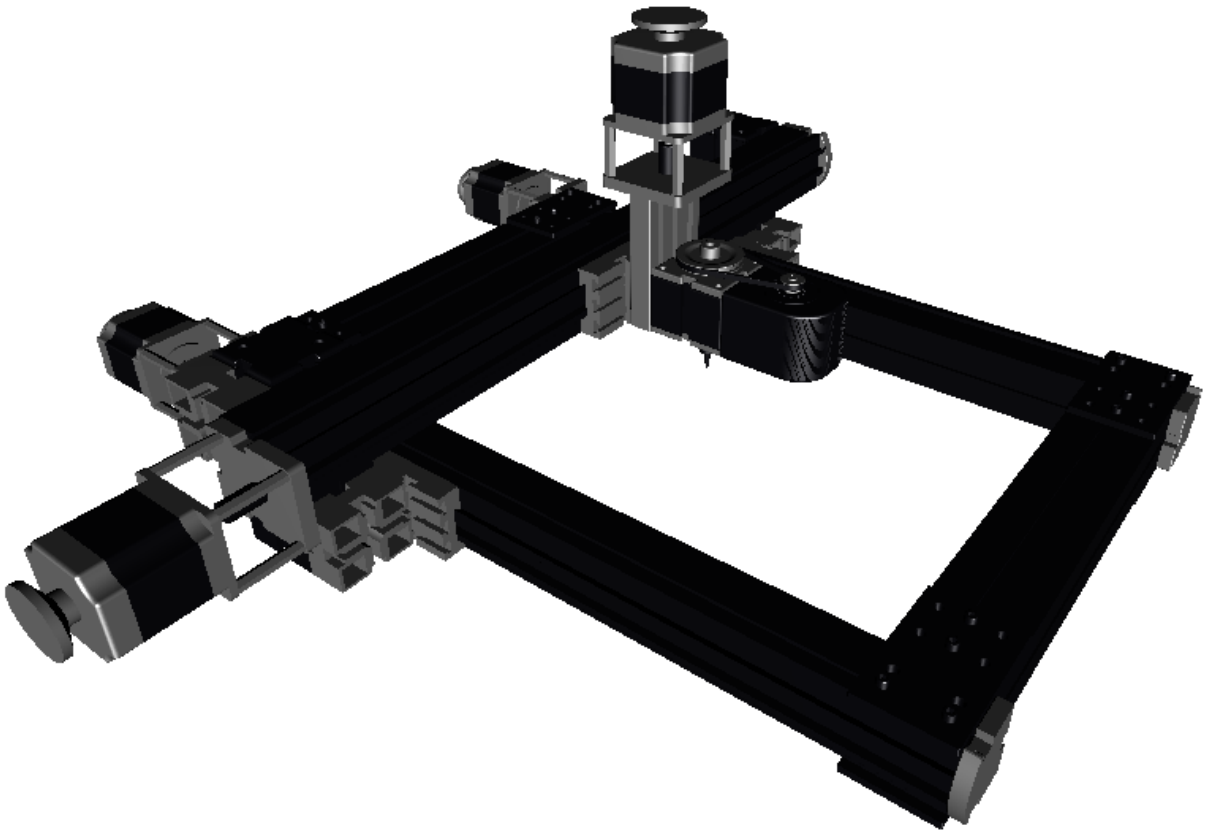
Obr. 47 3D Model - Rám s frézkou

- Frézku posuvnou v osách Y (1. Skupina) a Z (2. Skupina)



Obr. 48 3D Model - Rám s frézkou

Po správném importu, seřazení, nastavení prvků do skupin a pojmenování, můžeme celou soustavu exportovat již do animačního souboru *.3DS, vhodného pro systém Control Web 6.



Obr. 49 Animační model v prostředí Control Web 6

V systému Control Web 6 pak už pouze stačí navázat datové proměnné, definující posuv jednotlivých skupin modelu na tyto skupiny a určit osu, respektive směr pohybu.

6 Zhodnocení

Cílem této diplomové práce bylo vytvořit systémový návrh laboratorní úlohy, která by umožňovala polohování CNC stroje podle hodnot uložených v tabulce a generovaných programem EAGLE. Z tohoto důvodu jsem se detailně seznámil a popsal používané programy EAGLE a Control Web 6. Program EAGLE zde sloužil jako software pro návrh desky plošných spojů. Program Control Web 6 naopak jako nástroj pro vytvoření monitorovací, vizualizační a parametrizační aplikace. Aplikace vytvořená v systému Control Web 6, která měla za úkol komunikovat s programem EAGLE prostřednictvím DDE komunikace a data získané z programu EAGLE, konkrétně informace o poloze jednotlivých součástek na desce plošných spojů, převádět pomocí vytvořených skriptů na přesné souřadnice [X,Y] pro polohování stolního modelu CNC stroje. Získané souřadnice vrtacích děr jednotlivých součástek pak byly prostřednictvím OPC komunikace odesílány do PLC, které s využitím hardwarového spínacího obvodu ovládalo model CNC stroje.

V první fázi návrhu této laboratorní úlohy se bylo nutné seznámit s programem EAGLE, respektive s jeho možnostmi jak vyexportovat důležité informace o navržené desce plošných spojů. Program EAGLE umožňuje několik variant exportu do výstupního souboru, které se liší jak typem výstupního souboru, tak informacemi v něm obsaženými. Jako optimální řešení pro tuto úlohu se nakonec ukázalo využití exportu ve formě tzv. Partlistu, který umožňuje zapsat do souboru o zvoleném formátu informace o umístěných součástkách. Jedná se o označení součástky, název pouzdra dané součástky, pozici [X,Y] referenčního bodu součástky vůči referenčnímu bodu desky a natočení součástky kolem svého referenčního bodu o daný úhel. Jako formát výstupního souboru jsem zvolil sešit aplikace MS Excel, který je vhodný pro DDE komunikaci. Nevýhoda tohoto řešení se projevila při otevření vyexportovaného souboru, respektive při zjištění, že všechny informace o dané součástce se zapisují do jedné buňky tabulky, tj. do jednoho textového řetězce, odděleného mezerami. Bylo tedy nutné vymyslet a realizovat způsob, jak tyto data separovat a uložit do proměnných o vhodných datových typech. Tento proces se již prováděl ve vizualizační aplikaci systému Control Web 6. Všechny textové řetězce z daného Partlistu byly prostřednictvím DDE komunikace zkopírovány do Control Webu, aby již běh dané aplikace nebyl zatěžován udržováním DDE komunikace. Tyto textové řetězce byly zkopírovány do proměnných datových typů textových polí a následně zabudovanými procedurami systému rozkouskovány rovněž do polí proměnných, ale tentokráte už o potřebných datových typech. Název součástky do textového typu, pozice [X,Y] do reálného typu atp. Jakmile jsem takto získal všechny potřebné proměnné pro jednotlivé součástky, bylo možné s nimi dále pracovat.

V dalším kroku bylo nutné navrhnout a realizovat způsob, jakým by bylo prováděno vypočítávání souřadnic [X,Y] pájecích otvorů pro jednotlivé součástky. Vycházel jsem ze získaných souřadnic referenčního bodu součástky vůči referenčnímu bodu desky. Vzhledem k tomu, že každá součástka má svůj referenční bod umístěn na jiném místě, nebylo možné vytvořit algoritmus, který by se všemi součástkami pracoval stejně. Bylo nutné nahlížet na každou součástku individuálně, takže výpočtová část algoritmu byla pro každou součástku jiná. Celý algoritmus byl uzavřen do iterační smyčky, takže proběhl právě tolikrát, kolik součástek bylo obsažených na desce. V první části algoritmu docházelo k testování názvu pouzdra součástky, následně podle tohoto pouzdra se předaly proměnné, jiné pro každou součástku, se kterýma se dále pracovalo ve výpočtové části algoritmu. Všechny tyto výpočtové procesy běžely na pozadí aplikace v systému Control Web 6. Součástí této aplikace byl rovněž animační 3D model CNC stroje, grafické zobrazení desky plošných spojů a panel pro nastavení referenční polohy stroje. Toto nastavení referenční polohy bylo velmi důležité pro samotné odvrtávání. Nabízely se dvě varianty, jak provést nastavení referenční polohy fyzické desky na CNC modelu. První variantou bylo nastavit fyzickou desku ručně podle polohy frézky CNC modelu. Druhou variantou přesně naopak, softwarově přepočítat souřadnice jednotlivých pájecích otvorů dané desky podle pozice frézky CNC modelu. Pro tyto účely byl vytvořen algoritmus, který nejprve vybral dva referenční body desky, jeden nejvíce vlevo a nejvíce nahoře, druhý nejvíce vpravo a nejvíce dole. Nejprve bylo nutné najet frézou pomocí ručního ovládání do pozice prvního referenčního bodu na fyzické desce. Přesnost nastavení pozice do tohoto bodu byla rozhodující, protože byl tento bod brán jako počátek [0,0] a od tohoto bodu vypočítávány ostatní body. Následně bylo třeba najet stejným způsobem, co nejpřesněji, do pozice druhého, pravého referenčního bodu. Během tohoto pohybu byla zaznamenávána pozice nového bodu, který tímto vznikl. Souřadnice levého softwarového i fyzického bodu byly brány jako totožné. Tím, že jsme do pozice druhého referenčního bodu najížděli ručně a jeho pozici tak v podstatě vytvořili, tato pozice se nemohla shodovat v přesnosti se softwarovou. Vznikly tak dva referenční vektory se společným levým bodem. Mezi těmito vektory vznikl úhel, o který bylo nutné všechny body otočit kolem levého referenčního bodu. S využitím dalších algoritmů se zjišťovala orientace otočení a následně se přepočítaly nové pozice všech ostatních bodů. Tyto souřadnice pak bylo nutné přenést do PLC. Vzhledem k tomu, že jednotlivé souřadnice vrtacích otvorů se ukládaly do polí proměnných, bylo lepší celé toto pole přenést do PLC, aby vrtací proces nebyl závislý na OPC komunikaci. Po přenesení všech souřadnic do PLC se pak prostřednictvím OPC komunikace prováděly základní povely start, pauza, stop a PLC odesílalo informace o své aktuální pozici a počtu již odvrtaných děr.

Krokové motory, použité na modelu stolního CNC, byly ovládány sekvencí pulsů přivedených na jednotlivé cívky motoru a počet těchto pulsů byl známý. Bylo tudíž možné při známém počtu vyslaných pulzů, respektive známém počtu provedených otáček, získat informace o aktuální poloze v jednotlivých osách, při daném stoupání šroubovice. Hardwarové a softwarové možnosti celého systému neumožňovaly časování ovládací aplikace rychleji než 5ms. Takovéto časování umožnilo odeslat na krokový motor 200 pulsů za vteřinu, což u použitých krokových motorů přestavuje 1 otáčku za vteřinu. Rychlost lineárního posuvu byla bohužel velmi malá, 1,25mm/s, naopak při takovéto rychlosti otáčení produkuje krokový motor vysoký točivý moment, takže nehrozí možnost prokluzu krokového motoru. V případě, že by došlo k prokluzu krokového motoru, došlo by ke znehodnocení celé desky.

V současné době je laboratorní úloha navržena na odvrtávání děr desek plošných spojů. Vhodným rozšířením by bylo možné CNC stroj využít jako frézku, umožňující frézování požadovaných tvarů z desek různých materiálů.

7 Summary

The purpose of this work was to develop the system design of a laboratory task, which would allow positioning of the desktop CNC machine model, according to data stored into a table and generated by the EAGLE software. For this reason, it was necessary to become familiar with the EAGLE and the Control Web 6 software. The EAGLE software was used here as a tool for PCB designing. The monitoring and visualisation application was made using the Control Web 6 software.

In the first phase of this work it was necessary to find a way how to export important information about the designed PCB. The EAGLE software allows several variants of data export. These variants differ among themselves in the type of output file format and in information contained therein. As the optimal solution for this task turned out usage of the export in the form of so-called Partlist, that allows to write information about each PCB part into selected output file format. Most suitable for this task was a MS Excel output format. Minor disadvantage of this solution became apparent after open of the file. Individual information about the PCB parts (part package name, part position, part rotation) was placed into one text string, separated by spaces. Therefore it was necessary to create an algorithm, which would separate this information into data variables in appropriate data types. The final stage before we could get accurate drilling points coordinates of each part was to create algorithm for this purpose. All designed and created algorithms ran on the Control Web 6 application background. Communication between the Control Web 6 software and the MS Excel software was provided by the DDE communication protocol.

For the purpose of monitoring and visualisation of the laboratory task I had to create special application in the Control Web 6 software. As I mentioned above, this application contained scripts and computational algorithm, but the aim of this application was also to provide actual information about the CNC machine model movement and also to communicate with the PLC using the OPC communication protocol.

Currently, the laboratory task is designed for PCB soldering points drilling. Using suitable extension it would be possible to extend this laboratory task for parametric milling of different shapes.

Seznam použité literatury

1) BALÁTĚ, J. *Automatické řízení.* 1. vydání. Praha: BEN, 2003. 664 s. ISBN 80-7300-020-2.

2) BÍLÝ, R., CAGAŠ, P. AJ. *Control Web 2000. Průvodce systémem pro tvorbu a nasazení aplikací reálného času.* 1. vydání. Praha: Computer Press, 1999. 382 s. ISBN 80-7226-258-0.

3) CONTROL WEB 5. *Manuál.* Alcor – Moravské přístroje, a.s.,2007. [online] Dostupné z www: URL : <http://www.mii.cz/>.

4) CONTROL WEB 6. *Manuál.* Alcor – Moravské přístroje, a.s.,2010. [online] Dostupné z www: URL : <http://www.mii.cz/>.

5) DOKUMENTACE SCADA SYSTÉMU PROMOTIC, MICROSYS, spol. s.r.o. [online] Dostupné z www: URL :<http://www.promotic.eu/cz/pmdoc/Subsystems/Comm/DDE/DDE.htm>.

6) FARANA, R., SMUTNÝ, L. & VÍTEČEK, A. *Zpracování odborných textů z oblasti automatizace a informatiky.* Ostrava 2001, 68 s. ISBN 80-7078-737-6.

7) JANEČEK, J. 1993. *Distribuované systémy.* Praha : Vydavatelství ČVUT, 1993.

8) KEBO, V., LANDRYOVÁ, L. & HOLUB, P. *Návrh procesních systémů v prostředí SCADA/MMI - CITECT.* Ostrava: VŠB-TUO 1996. 100 s. ISBN 80-7078-410-5.

9) KOUDELA, T. *Návrh a realizace robustních algoritmů řízení pro vybrané technologické procesy.* Ostrava: VŠB-TUO, kat. ATŘ-352, 2005. 90 s. Diplomová práce.

10) LANDRYOVÁ, L., PAWELEK, M. & KONEČNÝ, M. *Návrh procesních systémů. Programové systémy SCADA/MMI.* 1. vyd. Ostrava, KAKI 1996, 96 s. ISBN 80-02-01100-7.

11) MARTINÁSKOVÁ, M. *Programovací jazyky pro PLC*. Časopis Automatizace 6/2004, strana 380. [online] Dostupné z www: URL : < http://www.automatizace.cz/article.php?a=142>.

12) MARTINEK, R. 2004. *Senzory v průmyslové praxi. 1. vyd.* Praha : BEN, 2004. 200s. ISBN 80-7300-114-4.

13) MATUŠEK, R. *Monitorování plošiny simulátoru vrtulníku v prostředí Control Web 5.* Ostrava: VŠB-TUO, kat. ATŘ-352, 2006. 83 s. Diplomová práce.

14) MORAVSKÉ PŘÍSTROJE. *Informační webový systém firmy.* [online] Dostupné z www: URL : <http://www.mii.cz/>.

15) NACHTIGAL CH. L. *Instrumentation and Control - Fundamentals and Applications. New York :* John Wiley & Sons, Inc. 1993.

16) ŘEZÁČ, K. *Krokové motory*. 2002 . [online] Dostupné z www: URL : < http://robotika.cz/articles/steppers/cs>

17) SPÍRAL, L. - PRŮŠA, J. 1993. *Logické objekty a řízení II - programování logických automatů*. Plzeň: ZČU 1993. 131 s.

18) ŠKUTA, J., REČKA, D., HAVLÍČEK, M. *aj. Komunikace řídicích systémů - prostřednictvím sítě Internet. Závěrečná zpráva grantového projektu FR VŠ MŠMT G1/0281/2001. Ostrava, VŠB TU Ostrava, 2001. 33 s.*

19) SMUTNÝ, L.,KLEČKA , R. *Prostředky automatického řízení 1*. [online] Dostupné z www: URL : <http://www.e-automatizace.cz/ebooks/ridici_systemy_akcni_cleny/Akc_el.html>.

20) VLACH, J. & VLACHOVÁ, V. 2000. *Počítačová rozhraní - přenos dat a řídící systémy.* Praha : BEN, 2000. 176s. ISBN 80-7300-010-5.

21) VLACH, J. *Řízení a vizualizace technologických procesů.* Praha: BEN, 1999, 160 s. ISBN 80-86056-66-X.

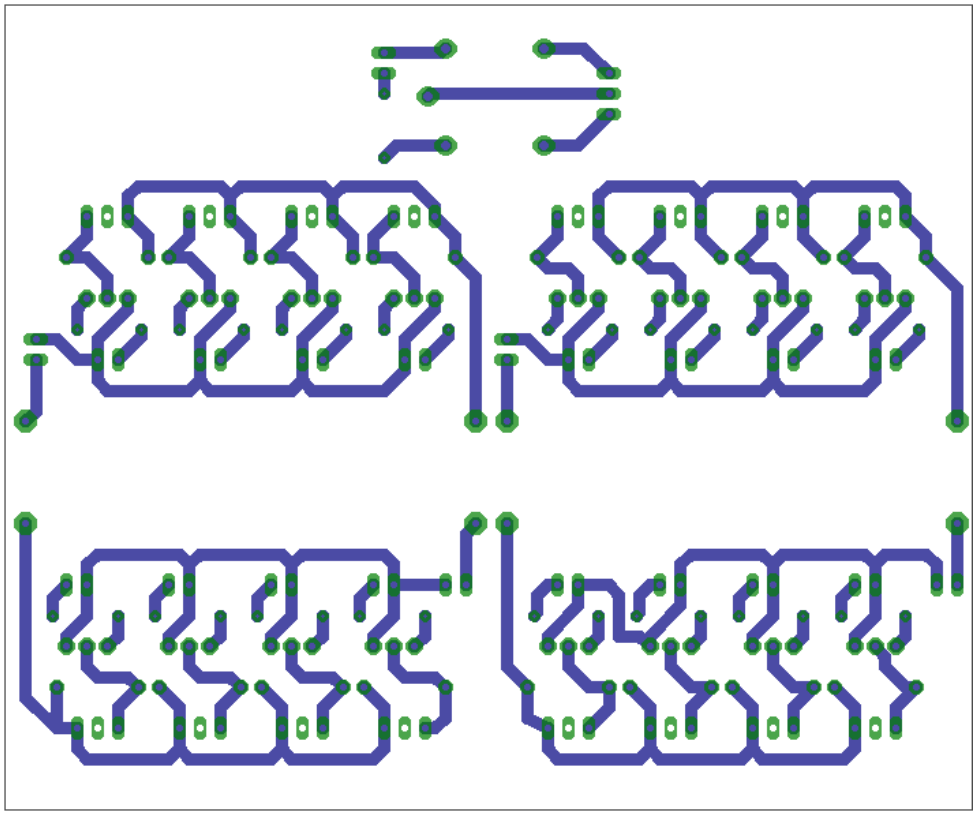
22) ŽLEBEK, R. *Počítačové řízení rozsáhlého laboratorního modelu s využitím operátorského vizualizačního systému.* Ostrava: VŠB-TUO, kat. ATŘ-352, 1996. 62 s. Diplomová práce.

Přílohy

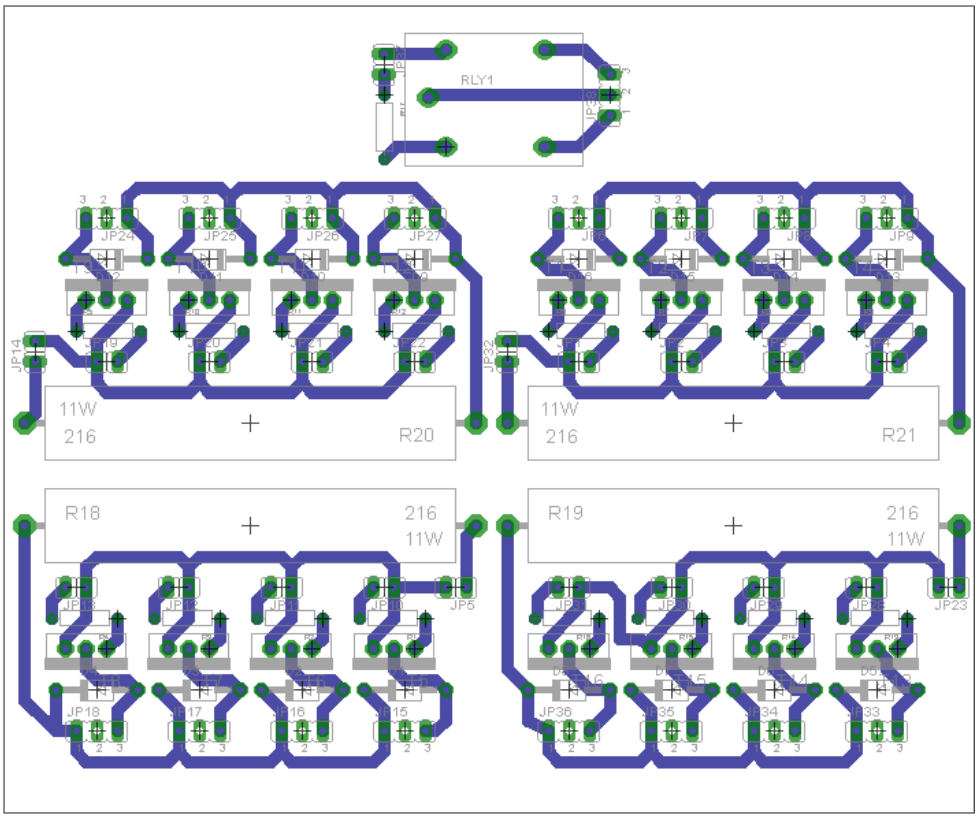
1) Schéma plošného spoje pro spínací obvod.
2) Část programového kódu pro výpočet souřadnic vrtacích děr pro několik vybraných součástek.
3) Programový kód pro separaci textových řetězců s daty součástek.

Příloha 1

Schéma plošného spoje pro spínací obvod



Obr. 1 Plošný spoj bez označení součástek



Obr. 2 Plošný spoj s označením součástek

Příloha 2

Část programového kódu pro výpočet souřadnic vrtacích děr pro několik vybraných součástek

```
for iP3=1 to ipart do

switch DDE_PACK[iP3] of

  case 'ARK500/2','ARK505/2','CK','E100','E130';
      pitch1=5.08;
      goto DRILL2;
  case 'DIL-16' ;
      pitch1=8.89;
      pitch2=3.81;
      drl=16;
      goto DRILL4;

…

DRILL2:
      switch DDE_ORNT[iP3] of
      case 'R0' , 'MR180';
      DRILL_X[idr]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr]=DDE_POSY_REAL[iP3];
      DRILL_X[idr+1]=DDE_POSX_REAL[iP3]+pitch1;
      DRILL_Y[idr+1]=DDE_POSY_REAL[iP3];
      idr=idr+2;
      case 'R90' , 'MR90';
      DRILL_X[idr]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr]=DDE_POSY_REAL[iP3];
      DRILL_X[idr+1]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr+1]=(DDE_POSY_REAL[iP3]+pitch1);
      idr=idr+2;
      case 'R180' , 'MR0';
      DRILL_X[idr]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr]=DDE_POSY_REAL[iP3];
      DRILL_X[idr+1]=(DDE_POSX_REAL[iP3]-pitch1);
      DRILL_Y[idr+1]=DDE_POSY_REAL[iP3];
      idr=idr+2;
      case 'R270' , 'MR270';
      DRILL_X[idr]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr]=DDE_POSY_REAL[iP3];
      DRILL_X[idr+1]=DDE_POSX_REAL[iP3];
      DRILL_Y[idr+1]=(DDE_POSY_REAL[iP3]-pitch1);
      idr=idr+2;
      end;
      DRILL_PART[idr-2]=DDE_SGN[iP3];
      DRILL_PART[idr-1]=DDE_SGN[iP3];
      goto FINISH;

…
```

```
DRILL4:
      switch DDE_ORNT[iP3] of
      case 'R0' , 'MR0' , 'R180' , 'MR180';
       iDIL=1;
       DRILL_X[idr]=DDE_POSX_REAL[iP3]-pitch1;
       DRILL_Y[idr]=DDE_POSY_REAL[iP3]+pitch2;
       DRILL_PART[idr]=DDE_SGN[iP3];
      loop
        DRILL_X[idr+1]=DRILL_X[idr]+2.54;
        DRILL_Y[idr+1]=DRILL_Y[idr];
        DRILL_PART[idr]=DDE_SGN[iP3];
        idr=idr+1;
        if iDIL=(drl/2) then
         exit;
         else
         iDIL=iDIL+1;
        end;
       end;
       iDIL=1;
       DRILL_X[idr]=DDE_POSX_REAL[iP3]-pitch1;
       DRILL_Y[idr]=DDE_POSY_REAL[iP3]-pitch2;
       DRILL_PART[idr]=DDE_SGN[iP3];
       loop
        DRILL_X[idr+1]=DRILL_X[idr]+2.54;
        DRILL_Y[idr+1]=DRILL_Y[idr];
        DRILL_PART[idr]=DDE_SGN[iP3];
        idr=idr+1;
         if iDIL=(drl/2) then
          exit;
          else
          iDIL=iDIL+1;
         end;
        end;

…

FINISH:
end;
send table_1;
pause 1;
send P4_Serazeni;
end_procedure;
```

Příloha 3

Programový kód pro separaci textových řetězců s daty součástek

```
procedure OnActivate();
begin
pause 0.5;
iP2=1;
for iP2 = 1 to ipart do
    DDE_SGN[iP2]=item( DDE[iP2], ' ', 0 );
    DDE_POS[iP2]=item( DDE[iP2], '()', 1 );
    DDE_POSX[iP2]=item( DDE_POS[iP2], ' ', 0 );
    DDE_POSY[iP2]=item( DDE_POS[iP2], ' ', 1 );
    DDE_PACK[iP2]=item( DDE[iP2], ' ', 1 );
    DDE_ORNT[iP2]=item( DDE[iP2], ' ', 5 );
    DDE_POSX_REAL[iP2]=val( DDE_POSX[iP2], 10);
    DDE_POSY_REAL[iP2]=val( DDE_POSY[iP2], 10);
end;
pause 0.1;
end_procedure;
```